

Europäisches Patentamt
European Patent Office
Office européen des brevets



(11) EP 1 049 097 A1

(12)

EUROPEAN PATENT APPLICATION

(43) Date of publication:
02.11.2000 Bulletin 2000/44

(51) Int. Cl.⁷: G11B 27/034, G11B 20/12,
H04N 5/85, G11B 27/32

(21) Application number: 00303506.0

(22) Date of filing: 26.04.2000

SEP.	4	PCA	89,817
CORRESP. US/UK:			
COUNTRY PC			

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 27.04.1999 KR 9915098
30.06.1999 KR 9926050

(71) Applicant:
SAMSUNG ELECTRONICS CO., LTD.
Suwon-City, Kyungki-do (KR)

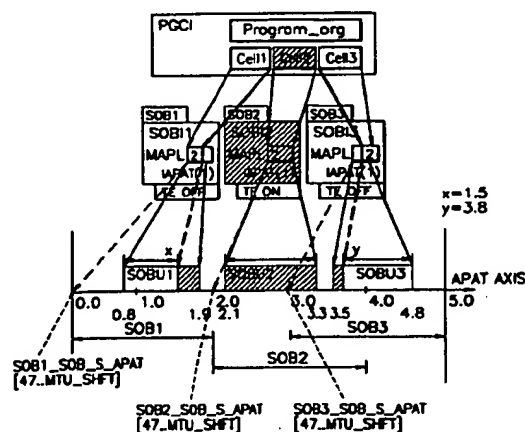
(72) Inventors:
• Park, Sung-woo
Seocho-gu, Seoul (KR)
• Moon, Seong-jin
Youngdeungpo-gu, Seoul (KR)
• Oh, Young-nam
Seongnam-city, Kyungki-do (KR)

(74) Representative:
Robinson, Ian Michael et al
Appleyard Lees,
15 Clare Road
Halifax HX1 2HY (GB)

(54) Method for temporarily deleting stream objects on a recording medium

(57) Methods for temporary erase, search, restoration and permanent erase of a stream object and a recording medium for storing additional information for restoration of divided stream objects are provided. In order to solve the problem that full restoration cannot be guaranteed in managing and editing (i.e. temporary deleting) a content which includes a film, music, or other data, when a plurality of units are used (SOB1, SOB2, SOB3), a method is provided for generating search information in dividing a stream object by temporary deletion. Using the information structure and restoration method provided here, stream objects divided by temporary deletion can be fully restored to their original state, and the temporarily-deleted part can be permanently deleted.

FIG. 6



Description

[0001] The present invention relates to editing and/or reproducing packet-structured data, and more particularly, to a method for temporary deletion, search, restoration, and permanent deletion of a stream object and a recording medium for storing additional information for restoration of the stream object divided by temporary deletion.

[0002] In general, a content is formed by a stream object (SOB), and an SOB is formed by a plurality of predetermined stream object units (SOBUs) and managed in an SOBU unit. At this time, the stream object can be, for example, data which are recorded while a user records something from the beginning to the end. More particularly, an episode of a miniseries or a film can be recorded in one stream object. Since a program can be used instead of a content, the word "program" will be used for explanation, hereinafter.

[0003] Figure 1 illustrates an example of the relationship between data and search information used in reading wanted data in an already-recorded stream object. When a user records a program, just one program can be recognized to the user's eyes, but a cell, which provides a meaningful search tool to the user, and stream object information (SOBI), which links information in a cell to actual data, exist internally.

[0004] Particularly as for a stream object, in order to improve the physical properties of recording media and the performance and/or management-easiness and efficiency of reproducing/editing devices, data are grouped in a predetermined unit (stream object unit, SOBU) for management, and information used in accessing an SOBU are stored in a mapping list (MAPL) of SOBI. The SOBU number begins from "1" and the MAPL can have a plurality of items. These items have information of increment application packet arrival time (IAPAT), which represents the time of SOBU's continuing.

[0005] In order to randomly access a program recorded in a storage device, search information on a program to be accessed is separately prepared and used, in general. As search information, information on data location in a program, program reproducing time, and program recording time are usually used. When a recorded program is a film, characteristic scenes can be used as search information. Here, a program recording time will be used as an example of search information.

[0006] Search information can have diverse formats and, generally data are grouped in a big bulk as a unit of searching object in order to reduce the volume of search information and enable fast search. Such bulk-grouping is usually done in a layered structure. Figure 1 illustrates an example of a three-layered structure.

[0007] When search information is made in the form of a layered structure, information on an upper layer includes information on a normal immediately-lower layer. As an upper layer in search information which is included in a program, a cell exists, below the cell, a stream object exists, and below the stream object, stream object units exist. Each layer of search information can have diverse relationships and, here, the following is assumed.

[0008] A program has one or more cells. A cell has one stream object. A stream object has one or more stream object units. A stream object unit has one or more data.

[0009] Figure 1 shows a case where Program_org is formed with Cell_org; Cell_org is formed with an SOB; the SOB is formed with SOBU1, SOBU2, and SOBU3, and each SOBU is formed with data. That is, Program_org has an information structure storing a cell it has, the cell has an information structure storing SOB's it has, and each SOB has a MAPL, the information structure storing SOBUs.

[0010] In addition, in order to show the range of stream object which search information manages, the first data arrival time of a cell (SC_S_APAT) and the last data arrival time of a cell (SC_E_APAT) are included in cell information, and the first data arrival time of a stream object (SOB_S_APAT) and the last data arrival time of a stream object (SOB_E_APAT) are included in stream object information.

[0011] A temporary erase (TE) flag represents that a stream object is temporarily deleted. Here, temporary deletion means that after a user deletes all or some parts of a program, a chance to cancel the deletion is given to the user. Unlike this, permanent deletion means a deletion which does not give a chance for cancelling the deletion.

[0012] Each item of MAPL are presented by incremental AP arrival time which means the contiguous time of each corresponding SOBU. Here, AP is an abbreviated form of application packet, and means packet-structured data. IAPAT, the contiguous time of an SOBU is defined as follows.

[0013] Search information expressed in time units has respective precision degree depending on application fields. As for moving picture experts group-2 (MPEG-2) system, time is counted and used in units of 27 MHz. In Figure 1, data is expressed in units of 0.1. In order to reduce the length of the search information MAPL, an SOB is assumed to express time in units of 1.0. When this is applied to a 48-bit register for expressing time, equal to or greater than point b18, shown in Figure 2, means integer part and less than point b18 means a decimal fraction. The location of b18, immediately upper decimal fractions, will be expressed as MTU_SHIFT.

[0014] In addition, each item of MAPL in Figure 1 has values of 3, 1, and 1, respectively, which are based on the following definition on MAPL. As a method for determining IAPAT of each SOBU, different methods are used according to the location of an SOBU in an SOB, that is, whether or not an SOBU is the last SOBU in an SOB.

[0015] For example, when *M* SOBUs exist in one SOB, the accumulated value of IAPATs from the first to the *i*-th

SOBU for i -th SOBU(i) except the last SOBU must not be greater than the first AP arrival time of SOBU($i+1$) by 1. IAPAT is assumed to be an integer expressed in units of 1.0, and the initial value of accumulation is assumed to be "0". This is expressed in the following equation 1.

$$5 \quad \text{SOBU_S_APAT}(i+1) \leq \text{SUM_IAPAT}(i) < \text{SOBU_S_APAT}(i+1) + 1$$

[0016] Here, SUM_IAPAT(i) represents the accumulated value of all preceding SOBU's IAPAT values, including the corresponding SOBU, that is, SOBU# i , and SOBU_S_APAT($i+1$) represents the arrival time of the first AP of SOBU($i+1$).

10 [0017] When M SOBUs exist, the accumulated value of IAPATs from the first to the M -th for SOBU(M), the last SOBU, must be greater than the arrival time of the last AP of SOBU(M) and must not be greater than that by 1. IAPAT is assumed to be an integer expressed in units of 1.0, and the initial value of accumulation starts from "0".

$$\text{SOBU_E_APAT}(M) < \text{SUM_IAPAT}(M) \leq \text{SOBU_E_APAT}(M) + 1$$

15 [0018] Here, SUM_IAPAT(M) represents the accumulated value of all preceding IAPATs, including the corresponding SOBU, that is, SOBU# M , and SOBU_E_APAT(M) represents the arrival time of the last AP of SOBU(M).

[0019] Referring to Figure 3, the concepts of the equations 1 and 2 will now be explained in detail.

20 [0020] In Fig. 3, as for SOBU1, the result of accumulation of SOBU1's IAPAT, that is, the value of SOBU1, must be equal to or greater than the arrival time of the first AP of SOBU2, and must not be greater than that by 1. That is, the result of accumulation must be equal to or greater than 1.9, and must be an integer less than 2.9, and therefore, the result must be 2. Accordingly, the IAPAT of SOBU1 is 2.

[0021] As for SOBU2, the result of accumulation of IAPATs of SOBU1 and SOBU2 must be the arrival time of the first AP of SOBU3, and must not be greater than that by 1. That is, the result of accumulation must be equal to or greater than 5.5, and must be an integer less than 6.5, and therefore, the result must be 6. Since The IAPAT of SOBU1 is 2, 25 The IAPAT of SOBU 2 is 4.

[0022] In this way, IAPATs of the first SOBU and a SOBU located in the middle can be obtained, and because a case in Figure 4A shows a boundary condition, it needs to be more carefully handled. When the IAPAT of SOBU2 of Figure 4A is calculated and when the arrival time of the first AP of SOBU3 is integer 5.0, the result of accumulation to 30 the IAPAT of SOBU2 is not 6 but 5.

[0023] In the meantime, as for SOBU6 in Figure 3, the result of accumulation of the integer part of the arrival time of the first AP of SOB and the IAPAT to SOBU6 must be greater than the arrival time of the last AP of SOBU6, and must not be greater than that by 1. That is, the result of accumulation must be greater than 10.8, and must be an integer less than or equal to 11.8, and therefore, it must be 11. Since the result of accumulation of IAPAT to SOBU5 is 10, the IAPAT 35 of SOBU6 is 1.

[0024] In this way, the IAPAT of the last SOBU in an SOB can be obtained, and because a case in Figure 4B shows a boundary condition, it needs to be more carefully handled. When the IAPAT of SOBU(6) of Figure 4B is calculated and when the arrival time of the last AP of SOBU(6) is an integer value of 11.0, the result of accumulation to the IAPAT of SOBU(6) is not 11 but 12.

40 [0025] In the meantime, when a program is temporarily deleted, this deletion is indicated by a temporary erase (TE) flag, which is promised and admitted in general. Therefore, when a part of a program is temporarily deleted, a stream object is divided into a temporarily deleted stream object and an undeleted stream object, and the TE flag is set in the temporarily deleted stream object. This is for restoring the deleted part later by changing only search information which links a user and data, without actually deleting data.

45 [0026] However, a method for generating search information in dividing a stream object has not been disclosed. Also, only the TE flag indicates that the temporarily deleted part has been temporarily deleted, which inevitably makes the TE flag reset in later restoration. Reading between those cells may be performed discontinuously because of the reset process.

[0027] If a restoration method having a process for resetting the TE flag is used, the restored program will have two or more cells. This means that the original program of Figure 1 is not restored fully because the operation between cells is not defined, though the temporarily deleted part is restored after the temporary deletion. Since from the standpoint 50 of a user, a restoration after temporary deletion means restoration to the original state, a method for fully restoring a stream object divided due to temporary deletion becomes required.

[0028] With the above problems in mind, it is an aim of the present invention to provide a method for temporary deletion, using search information for restoration of the original state, in a packet-structured stream object.

55 [0029] It is another aim to provide a method for high-speed searching, using search information, when a plurality of divided stream objects exist.

[0030] It is another aim to provide a method for full restoration of stream objects which were split into a plurality of

fragments during temporary deletion.

[0031] It is another aim to provide a method for updating the mapping list of a stream object, each mapping list corresponding to each boundary portion of each stream object divided by temporary deletion.

[0032] It is another aim to provide a method for updating the mapping list of a stream object, each mapping list corresponding to each boundary portion of each stream object, during the restoration of stream objects fragmented by editing to the original state.

[0033] It is another aim to provide a method for permanently deleting desired parts of a stream object.

[0034] It is another aim to provide a recording medium for storing additional information for restoration of stream objects which were split into a plurality of fragments during temporary deletion.

[0035] According to the present invention there is provided a method for temporarily deleting part of a stream object as set forth in claim 1 appended hereto. Also according to the present invention there is provided a method for searching a plurality of stream objects as set forth in claim 5 appended hereto. Further according to the present invention there is provided a method for restoring stream objects as set forth in claim 6 appended hereto. Further still, according to the present invention there is provided a method for updating mapping list information for stream object units, as set forth in claim 9 appended hereto. Yet further according to the present invention there is provided a method for updating mapping list information for a stream object unit as set forth in claim 12 appended hereto. Still further according to the present invention there is provided a method for permanently deleting some stream objects as set forth in claim 15 appended hereto. Also according to the present invention there is provided a recording medium storing search information for searching a plurality of stream objects, as set forth in claim 20 appended hereto. Preferred features and advantages of the invention will be apparent from the description and claims which follow.

[0036] According to a first aspect of the present invention there is provided a method for temporarily deleting part of a stream object recorded in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data and link information for linking search information to actual stream objects, the method comprising a step of: (a) updating search information and link information for a plurality of stream objects generated corresponding to ranges to be temporarily deleted and storing first additional information for indicating temporary deletion in a stream object corresponding to a temporary deletion range and second additional information for indicating that the stream object including the first additional information and the preceding stream object were one contiguous stream object before temporary deletion.

[0037] According to another aspect of the present invention there is provided a method for searching a plurality of stream objects stored in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method comprising the steps of: (a) setting the arrival time of a location a user wants to search, and setting the integer part of the arrival time of the first packet of a stream object included in the location, as the initial value of an accumulated value; (b) providing the accumulated value by accumulating the contiguous time of a stream object to the initial value; and (c) repeatedly performing the step (b) until the arrival time set for the location that a user wants to search is less than or equal to the accumulated value.

[0038] According to a third aspect of the present invention there is provided a method for restoring stream objects fragmented by editing in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method comprising a step of: (a) updating search information and link information in order to restore a plurality of stream objects generated corresponding to each editing range, to the original stream objects and nullifying first additional information for indicating that the corresponding stream object was edited, and second additional information for indicating that the edited stream object and the preceding stream object were one contiguous stream object before editing.

[0039] According to a fourth aspect of the present invention there is provided a method for updating each mapping list information for stream object units corresponding to the boundary of each stream object divided by temporary deletion, in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method comprising the steps of: (a) updating contiguous time related to the last stream object unit of the preceding stream object on the boundary of each stream object divided by temporary deletion; and (b) updating contiguous time related to the first stream object unit of the following stream object on the boundary of each stream object divided by temporary deletion.

[0040] According to a fifth aspect of the present invention, there is provided a method for updating mapping list information for a stream object unit corresponding to the boundary part of each stream object when restoring stream objects fragmented by editing in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method for updating each mapping list comprising the steps of: (a) updating contiguous time related to the last stream object unit of the preceding stream object on the boundary of each

stream object being integrated when stream objects are integrated; and (b) updating contiguous time related to the first stream object unit of the following stream object on the boundary of each stream object being integrated when stream objects are integrated.

5 [0041] According to a sixth aspect of the present invention there is provided a method for permanently deleting some stream objects in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method comprising the steps of: (a) updating search information for a stream object corresponding to a range to be permanently deleted; and (b) updating the link information, and updating each mapping list information for stream object units corresponding to the boundary of normal stream objects which are not permanently deleted.

10 [0042] According to a seventh aspect of the present invention there is provided a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects and storing a first additional information indicating that temporary deletion was performed in the stream object corresponding to the temporary deletion range, and a second additional information indicating that the stream object including the first additional information and the following stream object were one contiguous stream object before temporary deletion.

15 [0043] For a better understanding of the invention, and to show how embodiments of the same may be carried into effect, reference will now be made, by way of example, to the accompanying diagrammatic drawings in which:

20 Figure 1 illustrates an example of the relationship between data and search information used in reading a wanted data portion in general;

Figure 2 illustrates areas to be used in expressing application packet arrival time (APAT) and/or incremental application packet arrival time (IAPAT) in search information shown in Figure 1;

25 Figure 3 illustrates a process for obtaining IAPAT in general;

Figures 4A and 4B illustrate examples of obtaining IAPATs in boundary areas;

30 Figure 5 illustrates an example of determining range to be deleted during temporary deletion of part of a stream object;

Figure 6 illustrates an example of stream objects divided by temporary deletion;

35 Figure 7 is a table showing search information of each divided cell;

Figure 8 illustrates changes in search information of each cell before dividing and after dividing;

Figure 9 illustrates changes in the IAPAT of each cell before dividing and after dividing;

40 Figures 10A and 10B are flowcharts showing an IAPAT modification method related to the last stream object unit (SOBU) of the preceding stream object (SOB) when dividing a stream object;

45 Figures 11A and 11B are flowcharts showing an IAPAT modification method related to the first SOBU of the following SOB when dividing a stream object;

Figure 12 illustrates the values indicated by the variables used in the flowcharts shown in Figures 10A and 11A, when a stream object is divided into SOB1 and SOB2;

50 Figure 13 illustrates the values indicated by the variables used in the flowcharts shown in Figures 10B and 11B, when a stream object is divided into SOB2 and SOB3;

Figure 14 illustrates an example of restoring a stream object, which is divided by temporary deletion, using a simple restoration method;

55 Figure 15 illustrates an example of the structure of SOB information (SOBI) according to an embodiment of the present invention;

Figure 16 illustrates an embodiment of dividing stream object using the SOBI information shown in Figure 15;

Figure 17 illustrates an example of restoring stream object, using the SOBI information shown in Figure 15;

5 Figure 18 illustrates an example of the range to which the restoration process according to an embodiment of the present invention is applied;

Figure 19 illustrates a table showing search information related to stream objects before temporary deletion, after temporary deletion, and after complete restoration;

10 Figure 20 illustrates application packet (AP) arrival time of each SOB before integration and after integration;

Figure 21 illustrates relationships among stream object before and after integration, corresponding mapping list (MPAL), and IAPAT;

15 Figures 22A and 22B illustrate flowcharts showing an IAPAT modification method related to the last stream object unit of the preceding stream object when integrating stream object;

20 Figures 23A and 23B illustrate flowcharts showing an IAPAT modification method related to the first stream object unit of the following stream object when integrating stream object;

Figure 24 illustrates the values indicated by the variables used in the flowcharts of Figures 22A and 23A, when SOB1 and SOB2 are integrated into an SOB;

25 Figure 25 illustrates the value indicated by the variables used in the flowcharts of Figures 22B and 23B, when SOB2 and SOB3 are integrated into an SOB;

Figure 26 illustrates an example of permanent deletion of a part of a program;

30 Figure 27 illustrates an IAPAT modification method related to the last stream object unit of the preceding stream object when dividing a stream object by permanent deletion according to an embodiment of the present invention; and

35 Figure 28 illustrates an IAPAT modification method related to the first stream object unit of the following stream object when dividing a stream object by permanent deletion according to an embodiment of the present invention.

[0044] Hereinafter, embodiments of the present invention will be described in detail with reference to the attached drawings. The present invention is not restricted to the following embodiments, and many variations are possible within the scope of the present invention. The embodiments of the present invention are provided in order to more completely
40 explain the present invention to anyone skilled in the art.

[0045] Figure 5 illustrates an example of determining a range to be deleted during temporary deletion of a part of data. From a user's standpoint, it is desirable that a stored program can be handled randomly and to a minimum unit. That is, when the user wants to temporarily delete part of a program that is a film, the user would want to edit the film by scene units. At this time, since the degree of precision with which the user wants to delete is smaller than a stream
45 object unit (SOBU), which is a unit for an editing device to process data, the determination of a range to be deleted needs to be more carefully handled.

[0046] Figure 5 illustrates a user's partial deletion from 1.5 to 3.8 with a degree of precision of 0.1 which is a unit less than a SOBU. Using search information, the user finds time units predetermined for temporary deletion, and based on this, one cell is divided into three sub-cells. Figure 5 shows that a user temporarily deletes a time block which spans
50 xly, the middle point of the program, and deletion of both the first part and/or the last part of the program can be included.

[0047] In order to delete from x to y, first, the SOBU to which x belongs and the SOBU to which y belongs are obtained. Referring to Figure 5, x belongs to SOBU1 and y belongs to SOBU3. Therefore, in dividing one cell into three sub-cells, the first cell is formed by a stream object (SOB) that includes SOBUs before SOBU1, and SOBU1, the third
55 cell is formed by an SOB that includes SOBU3 and SOBUs after SOBU3, and the second cell is formed by an SOB that includes the remaining SOBUs. The reason is that the rule included in search information prescribes that a cell is formed by a stream object, and a stream object is formed by a plurality of SOBUs. Therefore, the first cell is formed by SOB1 that includes SOBU1, the third cell is formed by SOB3, which includes SOBU3, and the second cell is formed by

SOB2, which includes SOBU2.

[0048] Figure 6 illustrates an example of stream objects divided by temporary deletion. A block from x to y marked with hatched lines is Cell 2, the block to be deleted. In order to partially delete a part of an SOB, one cell information is divided into three cell information, and then a TE flag in the middle cell, which represents temporary deletion for later restoration, is set. The reason for setting the TE flag is in case the part where the temporary deletion is done is to be restored. At this time, the first cell includes a block from the beginning of the original cell to immediately before the block to be deleted, and the last cell includes a block from immediately after the block to be deleted to the end of the original cell. SOB information (SOBI) which links cell's search information with data can have a plurality of stream object units. Here, Cell 1 represents 0.8ix(=1.5), SOB1 represents 0.811.9, Cell 2 represents 2.113.3, SOB2 represents 2.1~3.3, Cell 3 represents 3.8(=y)14.8, and SOB3 represents 3.514.8.

[0049] The reason why the range of cell does not match the range of SOBI is that SOB records time in units of SOBU while a cell represents the time deleted by a user in the range of an SOB. By doing so, SOBI can manage an SOB in units of SOBU, and a cell can show the user whether or not user's manipulation is well handled.

[0050] Therefore, Figure 6 shows search information of a program which is divided into three cells through the temporary deletion process from x to y. By binding the block temporarily deleted, Cell 2 represents an information structure, cell, and sets the TE flag in order to show temporary deletion. When such search information is included, a reproducing device performs a reset process generally on the boundary of Cell 1 and Cell 3, when Cell 1 and Cell 3 are reproduced continuously. Since Cell 2 has been temporarily deleted, it is not reproduced.

[0051] In temporary deleting the time block between x and y, one cell is divided into three cells. Search information of each divided cell are shown in a table of Figure 7.

[0052] Application packet (AP) in the table of Figure 7 is data used in applications having a packet structure. Incremental AP arrival time (IAPAT) represents an AP arrival time in the form which can be recognized by accumulating AP arrival time; IAPAT(i) represents the IAPAT of SOBU(i); SOB_S_APAT (the arrival time of the first AP of an SOB) represents the arrival time of the first data located in the first point in an SOB; SOB_E_APAT (the arrival time of the last AP of an SOB) represents the arrival time of the last data located in the last point in an SOB; and SOB_S_SOBU (the first SOBU of an SOB) represents the number of the first SOBU located in the first point in an SOB. MAPL_ENT_Ns (the number of MAPL items) is the number of IAPATs included in an SOB, and is the same as the number of SOBUs; MAPL [IAPATs] (MAPL item) is a set of IAPATs; SOB_TY (SOB type) is the type of SOB, and includes the TE flag, an a contiguous (CT) flag, etc.; SC_S_APAT (the arrival time of the first AP of a cell) is the arrival time of the first data located in the first point in a cell; and SC_E_APAT (the arrival time of the last AP of a cell) is the arrival time of the last data located in the last point in a cell. Here, the CT flag will be explained in Figure 15 in detail.

[0053] The process for generating search information in temporary deletion will now be explained.

[0054] First, the SOBU number begins from 1. A value ix means the number of an SOBU which includes x. That is, SOBU(ix) refers to the SOBU which includes x. However, when x is the same as the arrival time of the first AP of an SOBU, ix is the number of an SOBU immediately before the SOBU which includes x. The value iy means the number of an SOBU which includes y. That is, SOBU(iy) refers to the SOBU which includes y. However, when y is the same as the arrival time of the last AP of an SOBU, iy is the number of an SOBU immediately after the SOBU which includes y. In Figure 6, therefore, SOBU(ix) is SOBU1, and SOBU(iy) is SOBU3. For reference, SOBU(ix+1) is SOBU2, and SOBU(iy-1) is SOBU2.

Step 1: The arrival time of the first AP of SOB_org is copied and then used as the arrival time of the first AP of SOB1.

Step 2: The arrival time of the first AP of SOBU(ix+1) becomes the arrival time of the first AP of SOB2.

Step 3: The arrival time of the first AP of SOBU(iy) becomes the arrival time of the first AP of SOB3.

Step 4: The arrival time of the last AP of SOBU(ix) becomes the arrival time of the last AP of SOB1.

Step 5: The arrival time of the last AP of SOBU(iy-1) becomes the arrival time of the last AP of SOB2.

Step 6: The arrival time of the last AP of SOB_org is copied and used as the arrival time of the last AP of SOB3. Examples of the steps 1 through 6 are shown in Figure 8.

Step 7: The first SOBU of SOB1 is set as the first SOBU of SOB_org.

Step 8: The first SOBU of SOB2 is set as SOBU(ix+1).

Step 9: The first SOBU of SOB3 is set as SOBU(iy).

When the steps 7 through 9 are applied to Figure 6, SOB1, SOB2, and SOB3 become the respective first SOBU because each of SOB1, SOB2, and SOB3 have one SOBU. Therefore, as shown in the table of Figure 7, the first SOBU of SOB1, SOB2, and SOB3 are SOBU1, SOBU2, and SOBU3, respectively.

Step 10: The number of MAPL items of SOB1 becomes ix.

Step 11: The number of MAPL items of SOB2 is determined as [SOBU(iy-1) number - SOBU(ix+1) number + 1].

Step 12: The number of MAPL items of SOB3 is determined as [the number of MAPL of an SOB - SOBU(iy) number + 1].

When the steps 10 through 12 are applied to Figure 6, the number of MAPL items of each of SOB1, SOB2, and SOB3 becomes 1. This means that after division into three parts after temporary deletion, each of SOB1, SOB2, and SOB3 have each one SOBU.

Step 13: The IAPATs of SOB1, SOB2, and SOB3 are determined. This will now be explained in relation to Figures 10A through 13.

When an SOB is divided, only the IAPAT value of an SOBU located on the boundary of division changes, since the rule for determining IAPAT is applied differently according to the SOBU location in an SOB. In Figure 9, the changed IAPAT after the SOB_org is divided into SOB1, SOB2, and SOB3 is shown. SOBU1 becomes the last SOBU of SOB1, and SOBU2 becomes the first SOBU of SOB2. In addition, SOBU2 becomes the last SOBU of SOB2, and SOBU3 becomes the first SOBU of SOB3. Therefore, the IAPAT of SOBU1 becomes 2.0, being reduced by 1.0, the IAPAT of SOBU2 becomes 2.0, being increased by 1.0, and the IAPAT of SOBU3 becomes 2.0, being increased by 1.0.

This will be explained in detail in relation to the flowcharts shown in Figures 10A through 11B. It is assumed that the arrival time of the last AP of SOBU(ix), the arrival time of the first AP of SOBU(ix+1), the arrival time of the last AP of SOBU(iy-1) and the arrival time of the first AP of SOBU(iy) are already known.

On the boundary between SOB1 and SOB2 after dividing the SOB by temporary deletion, the relative location of both the IAPAT related to SOBU(ix=1), which is the last SOBU of SOB1, and the IAPAT related to SOBU(ix+1=2), which is the first SOBU of SOB2, changes in an SOB. Therefore, these values must be modified.

The flowchart for showing a method for modifying the IAPAT of SOBU(ix), which is the last SOBU of SOB1, is shown in Figure 10A. First, in step S101, the arrival time of the first AP of SOBU(ix+1=2), which is the first SOBU of SOB2, is rounded up to an integer value and the integer is stored in a variable named preEnd_high (=3); the integer part of the arrival time of the last AP of SOBU(ix=1), which is the last SOBU of SOB1, is stored in a variable named preEnd_APAT_high(=1); and the IAPAT of SOBU(ix), which is the last SOBU of SOB1, is stored in a variable named preEnd_IAPAT(=3). In step S102, the difference (delta=2) between preEnd_high and preEnd_APAT_high (delta=2) is obtained, the difference is subtracted from preEnd_IAPAT(=3), and the result of the subtraction is incremented by a unit value (here, this is 1) in order to modify preEnd_IAPAT. In step S103, the modified preEnd_IAPAT (=2) is stored as the IAPAT of SOBU(ix). Therefore, the IAPAT of SOBU(ix=1) becomes 2.

Next, the method for modifying the IAPAT of SOBU(ix+1), which is the first SOBU of SOB2, is shown in Figure 11A. First, in step S121, the arrival time of the first AP of SOBU(ix+1=2), which is the first SOBU of SOB2, is stored in a variable named sucStart_APAT(=2.1), and the IAPAT of SOBU(ix+1), which is the first SOBU of SOB2, is stored in a variable named sucStart_IAPAT(=1). It is determined whether or not sucStart_APAT is an integer in step S122, and if sucStart_APAT is not an integer, then sucStart_IAPAT is modified by being incremented by a unit value (here, this is 1) in step S123. In step S124, the modified sucStart_IAPAT(=2) is stored as the IAPAT of SOBU(ix+1). When sucStart_APAT is an integer in the step S122, sucStart_IAPAT(=1), which is set in the step S121, is stored as the IAPAT of SOBU(ix+1) without modification. Therefore, the IAPAT of SOBU(ix+1=2) becomes 2.

Figure 12 is a schematic diagram of the values indicated by the variables used in the flowcharts of Figures 10A and 11A, when a stream object is divided into SOB1 and SOB2.

Similarly, also on the boundary between SOB2 and SOB3 when dividing SOB by temporary deletion, the IAPAT related to SOBU(iy-1=2), which is the last SOBU of SOB2, and the IAPAT related to SOBU(iy=3), which is the first SOBU of SOB3, changes in relative location in an SOB. These values must be modified as described in Figures 10B and 11B.

The flowchart for showing a method for modifying the IAPAT of SOBU(iy-1), which is the last SOBU of SOB2, is shown in Figure 10B. First, in step S111, the arrival time of the first AP of SOBU(iy=3), which is the first SOBU of SOB3, is rounded up to an integer value, and stored in a variable named preEnd_high (=4); the integer part of the arrival time of the last AP of SOBU(iy-1=2), which is the last SOBU of SOB2, is stored in a variable named preEnd_APAT_high (=3); and the IAPAT of SOBU(iy-1), which is the last SOBU of SOB2, is stored in a variable

named preEnd_IAPT (=2). In step S112, the difference ($\Delta=1$) between preEnd_high and preEnd_APAT_high ($\Delta=1$) is obtained; the difference is subtracted from preEnd_IAPT (=2); and the result of the subtraction is incremented by a unit value (here, this is 1) to modify preEnd_IAPT. In step S113, the modified preEnd_IAPT (=2) is stored as the IAPT of SOBU(iy-1). Therefore, the IAPT of SOBU(iy-1=2) becomes 2.

5 Next, the flowchart for showing a method for modifying the IAPT of SOBU(iy), which is the first SOBU of SOB3, is shown in Figure 11B. First, in step S131, the arrival time of the first AP of SOBU(iy=3), which is the first SOBU of SOB3, is stored in a variable named sucStart_APAT (=3.5), and the IAPT of SOBU(iy), which is the first SOBU of SOB3 is stored in a variable named sucStart_IAPT (=1), in step S131. It is determined in step S132 whether or not sucStart_APAT is an integer in step S132, and if sucStart_APAT is not an integer,
10 sucStart_IAPT (=1) is modified by being incremented by a unit value (here, this is 1) in step S133. The modified sucStart_IAPT (=2) is stored as the IAPT of SOBU(iy), or when sucStart_APAT is an integer in the step S132, sucStart_IAPT (=1), which is set in the step S131, is stored as the IAPT of SOBU(iy) without modification in step S134. Therefore, the IAPT of SOBU(iy=3) becomes 2.

15 Figure 13 illustrates the values indicated by the variables used in the flowcharts shown in Figures 10B and 11B, when a stream object is divided into SOB2 and SOB3.

Step 14: The TE flag and CT flag of SOB2, which are to be temporarily deleted, are set. The CT flag of SOB3, which is an SOB following the SOB2 to be temporarily deleted, is set.

20 Step 15: The arrival time of the first AP of a cell corresponding to SOB_org becomes the arrival time of the first AP of a cell corresponding to SOB1.

Step 16: The arrival time of the last AP of a cell corresponding to SOB1 becomes x.

25 Step 17: The arrival time of the first AP of a cell corresponding to SOB2 becomes the arrival time of the first AP of SOBU(ix+1).

Step 18: The arrival time of the last AP of a cell corresponding to SOB2 becomes the arrival time of the last AP of SOBU(iy-1).
30

Step 19: The arrival time of the first AP of a cell corresponding to SOB3 becomes y.

Step 20: The arrival time of the last AP of a cell corresponding to SOB_org becomes the arrival time of the last AP of a cell corresponding to SOB3.
35

[0055] The process in which an SOB is divided and SOBs are found using modified search information will now be described. Taking SOB3 as an example, expressions which can be applied to generalized cases are as follows.

[0056] For example, If data, that is an AP, which is located at 4.5 as shown in Figure 13, is to be searched, then the following steps are performed.

40 Step 1: 4.5 is stored in a target variable.

Step 2: The integer part of the arrival time of the first AP of an SOB (SOB_S_APAT) is taken as the initial value of accumulation.

45 Step 3: 1 is stored in variable i.

Step 4: accumulation of IAPT(i) is performed ($\text{sum} = \text{sum} + \text{IAPT}(i)$).

50 Step 5: If the target variable is less than or equal to the accumulated value (sum), then it indicates that the target variable is included in SOBU(i). If the target variable is greater than the accumulated value (sum), i is incremented by 1 and then the step 4 is performed.

[0057] Here, in finding an SOB that includes the wanted data, setting the integer part of the arrival time of the first data of an SOB as the initial value of accumulation, instead of setting the initial value to "0", enables IAPT to be kept at a smaller value, which is the merit of this method. If the initial value is set to "0" as in the previous methods, the IAPT of SOB3 must be 5, which is much greater than 1 of this method. This means a memory having a greater number of bits is required in implementing a circuit.

[0058] Figure 14 illustrates an example of restoring a stream object, which is divided by temporary deletion, using a simple restoration method, and an example of restoring the part that was temporarily deleted, using search information of prior art. The simple restoration nullifies the TE flag in an SOB which cannot be shown to a user. That is, in order to indicate that a cell is temporarily deleted, the TE flag in an SOBI is reset. When this method is used, a user can read all of cell 1, cell 2, and cell 3. However, to put it more concretely, since cell 1 can read the range between 0.8 and $x(=1.5)$, cell 2 can read the range between 2.1 and 3.3, and cell 3 can read the range between $y(=3.8)$ and 4.8, only those ranges can be read, while the ranges marked by deviant lines cannot be read. In addition, since a program which was formed by a cell is divided into three cells by partial deletion, definition of operations among cells are required when cells are restored. In order to solve the problem that reading among the cells is performed discontinuously because of a reset process inserted in the conventional method, the present invention provides an information structure of an SOB shown in Figure 15.

[0059] Figure 15 illustrates an example of implementing additional information according to the present invention, in the range of SOBI. Here, the additional information is referred to as contiguous (CT) flag. The CT flag and the TE flag can be stored in cell information.

[0060] When a CT flag indicating that divided SOBs are contiguous does not exist, the divided SOBs cannot be restored to the original state. Because, by using only the conventional search information, it cannot be known whether or not the neighboring SOBs in a divided state can be integrated. Therefore, information that indicates whether or not neighboring SOBs could be contiguously reproduced before division is necessary in order to fully restore the original state.

[0061] Therefore, though the CT flag can be a separate part of search information, or can exist in any location of existing search information, the CT flag is included in an SOBI in the embodiment of the present invention. This is because temporary deletion is performed on an SOB, and CT flag shows changes occurred in an SOB. Referring to Figure 15, therefore, the TE flag indicates temporary deletion and the CT flag indicates that divided SOBs were one contiguous SOB before temporary deletion.

[0062] The CT flag means that the SOB related to the CT flag and the preceding SOB were one contiguous SOB before temporary deletion. This means that the SOB related to the CT flag can be integrated with the preceding SOB. Such a case frequently occurs when dividing an SOB by partial deletion. Of course, both a CT flag indicating that a current SOB was contiguous with the preceding SOB and a CT flag indicating that a current SOB was contiguous with the following SOB can be included in one SOBI. This structure has a merit that information can be gathered and managed in one place, but it is less regular than a structure in which only information on the preceding SOB is included, and management thereof is inconvenient. Therefore, the structure in which only one CT flag exists in an SOB will be explained here.

[0063] Figure 16 illustrates an example of adding the SOBI of Figure 15 to divided SOBs. Unlike the example of Figure 6, a CT flag is added to each SOBI for an SOB divided by temporary deletion in the present invention. Since SOB2 of Cell2 represents the temporarily deleted part, the TE flag is set in SOBI2. In addition, since the preceding SOB1 and SOB2 were one contiguous SOB before temporary deletion, the CT flag is also set in SOBI2. Since SOB2 and SOB3 were one contiguous SOB before partial deletion, the CT flag is set in SOBI3, which related to SOB3. Because SOB3 was not temporarily deleted, the TE flag is not set in SOBI3.

[0064] Figure 17 illustrates an example of restoring divided SOBs shown in Figure 16, using the CT flag according to the present invention. Figure 17 shows that the ranges from x to 1.9 and from 3.5 to y , marked by *, which cannot be restored by the previous art of Figure 14, can be fully restored.

[0065] The restoration process according to the present invention will now be explained.

Step 1: SOBs to be the object of integration process carried out in a full restoration process is determined.

Here, for explanation, a cell which is temporarily deleted and to be restored is referred to as a target cell, and an SOB corresponding to the target cell is referred to as a target SOB. In Figure 16, a target cell is Cell 2, and the target SOB is SOB2.

To begin a full restoration process, first, SOBs to be integrated are determined. This process can be done, using the TE flag and the CT flag, and the range where a the restoration process is applied is shown in Figure 18. Full restoration begins from the target SOB. When the TE flag and the CT flag of the target SOB are set, the TE flag of the preceding SOB must be in a reset state, which means that the preceding SOB must be in a normal state.

When the normal state of flags of the two SOBs are confirmed, the SOBs can be integrated. In addition, when the CT flag is set and the TE flag is reset in the following SOB of the target SOB, the two SOBs can be integrated. In conclusion, it can be inferred that all of the preceding SOB, the target SOB, and the following SOB can be integrated. When the program shown in Figure 16 is fully restored, all of SOB1, SOB2, and SOB3 can be integrated using the CT flag as shown in Figure 18.

Step 2: The values of related search information in SOBI and cell information are restored to the values of the orig-

inal search information. Figure 19 illustrates a table showing search information related to SOB1 before temporary deletion, after temporary deletion and after full restoration.

[0066] A method for updating search information in restoration of an SOB will now be explained.

5

Step 1: The arrival time of the first AP of SOB1 becomes the arrival time of the first AP of SOB_rec.

Step 2: The arrival time of the last AP of SOB3 becomes the arrival time of the last AP of SOB_rec.

10

Step 3: The first SOBU of SOB1 becomes the first SOBU of SOB_rec. When SOB1, SOB2, and SOB3 are integrated, the first SOBU of SOB1 becomes the first SOBU, and Figure 20 explains the steps 1 through 3.

15

Step 4: The sum of the number of MAPL items of each of SOB1, SOB2, and SOB3 becomes the number of MAPL item of SOB_rec. Since each of SOB1, SOB2, and SOB3 has one SOBU in the embodiment of the present invention, shown in Figure 16, when SOB1, SOB2 and SOB3 are integrated, the number of SOBUs is the sum of SOBUs the SOB1 have.

Step 5: The IAPAT of SOB_rec is modified. This step will be explained, referring to FIGS 21 through 25.

20

Figure 21 illustrates SOB1 before and after integration and the IAPATs and MAPLs corresponding to respective SOB1. The IAPATs have different values depending on the locations of corresponding SOBUs. This is because the definition of IAPATs are different depending on the locations of SOBUs.

25

Before integration, SOBU1 is the last SOBU of SOB1, and SOBU2 is the first SOBU of SOB2. In addition, SOBU2 is the last SOBU of SOB2, and SOBU3 is the first SOBU of SOB3. Therefore, the IAPAT of SOBU1 is 3.0 incremented by 1.0, the IAPAT of SOBU2 is 1.0 decremented by 1.0 and the IAPAT of SOBU3 is 1.0 decremented by 1.0.

First, the last IAPAT of SOB1, the first and last IAPATs of SOB2, and the first IAPAT of SOB3 are modified. In Figure 21, since SOB2 is formed by one SOBU, the first SOBU of SOB2 and the last SOBU of SOB2 are the same.

30

After integration, on the boundary between SOB1 and SOB2, the first SOBU of SOB2, the IAPAT related to SOBU1, the last SOBU of SOB1, and the IAPAT related to SOBU2, the first SOBU of SOB2, change in the relative locations inside the SOB. Therefore, the values must be modified.

35

First, the flowchart for showing a method for modifying the IAPAT of SOBU1, the last SOBU of SOB1, is shown in Figure 22A. First, in step S201, the integer part of the arrival time of the last AP of SOBU1, the last SOBU of SOB1, is incremented by 1, and then stored in a variable named preEnd_high (=2); the arrival time of the first AP of SOB2 is stored in a variable named sucStart_APAT (=2.1); the integer part of the arrival time of the first AP of SOB2 is stored in sucStart_APAT_high (=2.0); and the IAPAT of SOBU1, the last SOBU of SOB1, is stored in a variable named preEnd_IAPAT (=2). After obtaining the difference (delta=0) between the sucStart_APAT_high and preEnd_high, the difference (delta) is added to preEnd_IAPAT (=2) in step S202. It is determined in step S203 whether or not sucStart_APAT is an integer. When sucStart_APAT is not an integer, preEnd_IAPAT (=2) is modified by being incremented by 1 in step S204. The modified preEnd_IAPAT (=3) is stored as the IAPAT of the last SOBU of SOB1, or when sucStart_APAT is an integer in the step S203, preEnd_IAPAT (=2), which is obtained in the step S202, is stored as the IAPAT of the last SOBU of SOB1 without modification in step S205. Therefore, the IAPAT of SOBU1 is 3.

40

Next, the flowchart for showing a method for modifying the IAPAT of SOBU2, the first SOBU of SOB2 after integration, is shown in Figure 23A. First, in step S221, the arrival time of the first AP of SOBU2, the first SOBU of SOB2, is stored in a variable named sucStart_APAT (=2.1), and the IAPAT of SOBU2, the first SOBU of SOB2, is stored in a variable named sucStart_IAPAT (=2). It is determined whether or not sucStart_APAT is an integer in step S222. When sucStart_APAT is not an integer, sucStart_IAPAT (=2) is modified by being decreased by 1 in step S223. The modified sucStart_IAPAT (=1) is stored as the IAPAT of SOBU2, or when sucStart_APAT is an integer in the step S222, sucStart_IAPAT (=2), which is set in the step S221, is stored as the IAPAT of the first SOBU of SOB2 without modification in step S224. Therefore, the IAPAT of SOBU2 is 1.

50

Figure 24 illustrates the values indicated by the variables used in the flowcharts of Figures 22A and 23A when SOB1 and SOB2 are integrated into an SOB.

Likewise, after integration, on the boundary between SOB2 and SOB3, the relative locations of the IAPAT related to SOBU2, of the last SOBU of SOB2, and of the IAPAT related to SOBU3, the first SOBU of SOB3, change inside the SOB. Therefore, the values must be modified.

55

First, the flowchart for showing a method for modifying the IAPAT of SOBU2, the last SOBU of SOB2, is shown in Figure 22B. First, in step S211, the integer part of the arrival time of the last AP of SOBU2, the last SOBU of SOB2, is incremented by 1, and then stored in a variable named preEnd_high (=4); the arrival time of the first AP

of SOB3 is stored in a variable named sucStart_APAT (=3.5); the integer part of the arrival time of the first AP of SOB3 is stored in sucStart_APAT_high (=3.0); the IAPAT of SOBU2, the last SOBU of SOB2, is stored in a variable named preEnd_IAPAT (=1). After obtaining the difference ($\Delta = -1$) between the sucStart_APAT_high and preEnd_high, the difference (Δ) is added to preEnd_IAPAT (=1) in step S212. It is determined in step S213 whether or not sucStart_APAT is an integer. When sucStart_APAT is not an integer, preEnd_IAPAT (=1) is modified by being incremented by 1 in step S214. The modified preEnd_IAPAT is stored as the IAPAT of the last SOBU of SOB2, or when sucStart_APAT is an integer in the step S213, preEnd_IAPAT (=0), which is set in the step S212, is stored as the IAPAT of the last SOBU of SOB2 without modification in step S215. Therefore, the IAPAT of SOBU1 is 1.

Next, the flowchart for showing a method for modifying the IAPAT of SOBU3 is shown in Figure 23B. First, the arrival time of the first AP of SOBU3, the first SOBU of SOB3, is stored in a variable named sucStart_APAT (=3.5), and the IAPAT of SOBU3, the first SOBU of SOB3 is stored in sucStart_IAPAT (=3) in step S231. It is determined in step S232 whether or not sucStart_APAT is an integer. When sucStart_APAT is not an integer, sucStart_IAPAT (=2) is modified by being decreased by 1 in step S233. The modified sucStart_IAPAT (=1) is stored as the IAPAT of SOBU3, or when sucStart_APAT is an integer in the step S232, sucStart_IAPAT (=2), which is set in the step S231, is stored as the IAPAT of SOBU3 without modification in step S234. Therefore, the IAPAT of SOBU3 is 1.

Figure 25 illustrates the values indicated by the variables used in the flowcharts of Figures 22B and 23B, when SOB2 and SOB3 are integrated into an SOB.

Step 6: The status of SOB_rec is determined. Since the SOB is fully restored, the TE flag and the CT flag of SOB_rec are reset.

Step 7: The arrival time of the first AP of a corresponding cell in SOB1 becomes the arrival time of the first AP of a corresponding cell in SOB_rec.

Step 8: The arrival time of the last AP of a corresponding cell in SOB3 becomes the arrival time of the last AP of a corresponding cell in SOB_rec.

[0067] Figure 26 illustrates an example of permanent deletion of the range from x to y in an SOB. Here, the SOBU which is permanently deleted is SOBU2. In permanent deletion, link information as well as search information must be changed. Therefore, in permanent deletion of the range from x to y, since SOBU1 is the last SOBU of the newly-generated SOB by division, the corresponding IAPAT must be changed, and since SOBU3 is the first SOBU of the newly-generated SOB by division, the corresponding IAPAT must be changed. In addition, search information of SOBU2, the SOBU located between x and y that can be fully deleted, must be deleted. Link information related to SOBU2 must be deleted, too. For reference, when only a part of SOBU is deleted, related search information is kept.

[0068] When search information and link information related to SOBU2 is fully deleted like this, search information and link information of Program_org as shown in Figure 26 are updated as search information of Cell1 and Cell 2.

[0069] Figure 27 illustrates a flowchart for showing a method for modifying the IAPAT related to the last SOBU of the preceding SOB (corresponding to Cell 1 of Figure 26), when dividing an SOB by permanent deletion according to the present invention. After the division of an SOB by permanent deletion, since on the boundary between SOB1 and SOB2, the relative location of SOBU(ix=1), the last SOBU of SOB1, changes inside the SOB, the value must be modified. It is assumed that the arrival time of the last AP of SOBU(ix), the arrival time of the first AP of SOBU(ix+1), the arrival time of the last AP of SOBU(iy-1), and the arrival time of the first AP of SOBU(iy) are known already.

[0070] First, in step S301, the arrival time of the first AP of SOBU(ix+1=2), the first SOBU of SOB2, is rounded up to an integer value and stored in a variable named preEnd_high (=3); the integer part of the arrival time of the last AP of SOBU(ix=1), the last SOBU of SOB1, is stored in a variable named preEnd_APAT_high (=1); and the IAPAT of SOBU(ix), the last SOBU of SOB1, is stored in a variable named preEnd_IAPAT (=3). In step S302, the difference ($\Delta = 2$) between preEnd_high and preEnd_APAT_high is obtained; the difference ($\Delta = 2$) is subtracted from preEnd_IAPAT (=3); and then preEnd_IAPAT is modified by increasing the result of the subtraction by 1. In step S303, the modified preEnd_IAPAT (=2) is stored as the IAPAT of SOBU(ix). Therefore, the IAPAT of SOBU(ix=1) is 2.

[0071] Figure 28 illustrates a flowchart for showing a method for modifying the IAPAT related to the first SOBU of the following SOB (corresponding to Cell 2 of Figure 26), when dividing an SOB by permanent deletion according to the present invention. Since the relative location of the IAPAT related to SOBU(iy=3), the first SOBU of SOB2, changes inside an SOB, the value must be modified.

[0072] First, the arrival time of the first AP of SOBU(iy=3), the first SOBU of SOB2, is stored in a variable named sucStart_APAT (=3.5), and the IAPAT of SOBU(iy), the first SOBU of SOB2, is stored in a variable named sucStart_IAPAT (=1) in step S311. It is determined whether or not sucStart_APAT is an integer in step S312. When sucStart_APAT is not an integer, sucStart_IAPAT (=1) is modified by being incremented by 1 in step S313. The modified

sucStart_IAPAT (=2) is stored as the IAPAT of SOBU(iy), or when sucStart_APAT is an integer in the step S312, sucStart_IAPAT (=1), which is set in the step S311, is stored as the IAPAT of SOBU(iy) without change in step S314. Therefore, the IAPAT of SOBU(iy=3) is 2.

[0073] The present invention can be used in a recording/reproducing device using digital recording media, and, in particular, it can be effectively used in a stream recorder.

[0074] According to the present invention, a method is provided for generating search information, which has not been suggested in the previous art, for use when an SOB is divided by temporary deletion. When the information structure and restoration method according to the present invention are used, SOBs divided by temporary deletion are fully restored to their original state.

[0075] In addition, in a process for searching an SOB including wanted data, the integer part of the arrival time of the first data of the SOB is set as the initial value, instead of setting the initial value at "0", which enables high speed searching.

[0076] Also, since in permanently deleting a part which is temporarily deleted, the range to be deleted can be found using only search information, it is appropriate for such applications that need fast deletion.

[0077] The reader's attention is directed to all papers and documents which are filed concurrently with or previous to this specification in connection with this application and which are open to public inspection with this specification, and the contents of all such papers and documents are incorporated herein by reference.

[0078] All of the features disclosed in this specification (including any accompanying claims, abstract and drawings), and/or all of the steps of any method or process so disclosed, may be combined in any combination, except combinations where at least some of such features and/or steps are mutually exclusive.

[0079] Each feature disclosed in this specification (including any accompanying claims, abstract and drawings), may be replaced by alternative features serving the same, equivalent or similar purpose, unless expressly stated otherwise. Thus, unless expressly stated otherwise, each feature disclosed is one example only of a generic series of equivalent or similar features.

[0080] The invention is not restricted to the details of the foregoing embodiment(s). The invention extend to any novel one, or any novel combination, of the features disclosed in this specification (including any accompanying claims, abstract and drawings), or to any novel one, or any novel combination, of the steps of any method or process so disclosed.

Claims

1. A method for temporarily deleting part of a stream object (SOB) recorded in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data (APAT) and link information for linking search information to actual stream objects, the method characterised by the step of:

(a) updating search information and link information for a plurality of stream objects (SOB1, SOB2, SOB3) generated corresponding to ranges to be temporarily deleted and storing first additional information (TE) for indicating temporary deletion in a stream object (SOB2) corresponding to a temporary deletion range and second additional information (CT) for indicating that the stream object including the first additional information and the preceding stream object were one contiguous stream object before temporary deletion.

2. The method of claim 1, wherein the first additional information (TE) and the second additional information (CT) are included in the link information.

3. The method of claim 1, wherein the first additional information (TE) and the second additional information (CT) are included in the search information.

4. The method of any of claims 1 to 3, wherein the step (a) comprises sub-steps of:

(a1) updating the arrival time of the first packet and the last arrival time of the packet of each stream object that is divided by temporary deletion;

(a2) updating the number of mapping list of each stream object that is divided by temporary deletion;

(a3) dividing stream object units forming each stream object that is divided by temporary deletion;

(a4) modifying contiguous time of stream object units on the boundary of each stream object that is divided by

temporary deletion;

(a5) setting first additional information (TE) and second additional information (CT) for a stream object (SOB2) corresponding to the range temporarily deleted and setting second additional information (CT) for a stream object (SOB3), a part of which is included in the range that is temporarily deleted and follows the stream object that corresponds to the range that temporarily deleted; and

(a6) updating the arrival time of the first packet and the last packet in a cell of a corresponding stream object divided by temporary deletion for every cell.

5. A method for searching a plurality of stream objects stored in a recording medium storing search information for searching a plurality of stream objects (SOB), each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method characterised by the steps of:

(a) setting the arrival time of a location a user wants to search, and setting the integer part of the arrival time of the first packet of a stream object included in the location, as the initial value of an accumulated value;

(b) providing the accumulated value by accumulating the contiguous time of a stream object to the initial value; and

(c) repeatedly performing the step (b) until the arrival time set for the location that a user wants to search is less than or equal to the accumulated value.

6. A method for restoring stream objects fragmented by editing in a recording medium storing search information for searching a plurality of stream objects (SOB), each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method characterised by the step of:

(a) updating search information and link information in order to restore a plurality of stream objects generated corresponding to each editing range, to the original stream objects and nullifying first additional information (TE) for indicating that the corresponding stream object was edited, and second additional information (CT) for indicating that the edited stream object and the preceding stream object were one contiguous stream object before editing.

7. The method of claim 6, wherein the editing is temporary deletion.

8. The method of claim 6 or 7, wherein the step (a) further comprises the sub-steps of:

(a1) updating the arrival time of the first packet and the arrival time of the last packet in an original stream object that is being integrated, with the arrival time of the first packet in a stream object including the first part of a temporarily deleted range, and the arrival time of the last packet in a stream object including the last part of a temporarily deleted range, respectively;

(a2) updating the number of mapping list items of the original stream object that is being integrated, with the sum of the numbers of mapping list items of each stream objects divided by temporary deletion;

(a3) integrating stream object units forming each stream object divided by temporary deletion;

(a4) modifying the value of contiguous time of each stream object unit of the original stream object that is being integrated;

(a5) resetting a first additional information and a second additional information for the original stream object that is being integrated; and

(a6) updating the arrival time of the first packet and the last packet in a cell corresponding to the original stream object that is being integrated, with the arrival time of the first packet in a cell including the first part of temporarily deleted range, and the arrival time of the last packet in a cell including the last part of temporarily deleted

range, respectively.

- 5 9. A method for updating each mapping list information for stream object units corresponding to the boundary of each stream object divided by temporary deletion, in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method characterised by the steps of:

10 (a) updating contiguous time related to the last stream object unit of the preceding stream object on the boundary of each stream object divided by temporary deletion; and

(b) updating contiguous time related to the first stream object unit of the following stream object on the boundary of each stream object divided by temporary deletion.

- 15 10. The method of claim 9, wherein the step (a) further comprises sub-steps of:

(a1) providing a first value obtained in the form of integer by rounding up the arrival time of the first packet of the first stream object unit of a stream object corresponding to a temporarily deleted range, and a second value obtained by taking the integer part of the arrival time of the last packet of the last stream object unit of the preceding stream object;

20 (a2) providing a first subtraction result by subtracting the second value from the first value, and providing a second subtraction result by subtracting the first subtraction result from the contiguous time of the last stream object unit of the preceding stream object; and

25 (a3) providing modified contiguous time related to the last stream object unit of the preceding stream object by incrementing the result of the second subtraction by the unit value.

11. The method of claim 9 or 10, wherein the step (b) further comprises the sub-steps of:

30 (b1) determining whether or not the arrival time of the first packet of the first stream object unit of the following stream object is an integer; and

35 (b2) when the arrival time of the first packet of the first stream object unit of the following stream object is not an integer, modifying contiguous time of the first stream object unit of the following stream object by incrementing by the unit value, and otherwise, providing the contiguous time of the first stream object unit of the following stream object, without modification.

- 40 12. A method for updating mapping list information for a stream object unit corresponding to the boundary part of each stream object when restoring stream objects fragmented by editing in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method characterised by the steps of:

45 (a) updating contiguous time related to the last stream object unit of the preceding stream object on the boundary of each stream object being integrated when stream objects are integrated; and

(b) updating contiguous time related to the first stream object unit of the following stream object on the boundary of each stream object being integrated when stream objects are integrated.

- 50 13. The method of claim 12, wherein the step (a) further comprises sub-steps of:

(a1) providing a first value obtained by adding a unit value to the integer part of the arrival time of the last packet of the last stream object of the preceding stream object, and providing a second value obtained by taking the integer part of the packet arrival time of the first stream object unit of the following stream object;

55 (a2) obtaining a difference of contiguous time of the last stream object unit of the preceding stream object, from the second value, and providing the result of sum by adding the difference to the contiguous time of the last stream object unit of the preceding stream object; and

(a3) when the packet arrival time of the first stream object unit of the following stream object is not an integer, providing contiguous time of the last stream object unit of the preceding stream object modified by incrementing the result of sum by the unit value, and otherwise, providing the contiguous time of the last stream object unit of the preceding stream object without change.

5 14. The method of claim 12 or 13, wherein the step (b) further comprises sub-steps of:

(b1) determining whether or not the arrival time of the first packet of the first stream object of the following stream object is an integer; and

10 (b2) when the arrival time of the first packet of the first stream object unit of the following stream object is not an integer, modifying contiguous time of the first stream object unit of the following stream object by decrementing it by the unit value, and otherwise, providing the contiguous time of the first stream object unit of the following stream object without modification.

15 15. A method for permanently deleting some stream objects in a recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects, the method characterised by the steps of:

20 (a) updating search information for a stream object corresponding to a range to be permanently deleted; and

(b) updating the link information, and updating each mapping list information for stream object units corresponding to the boundary of normal stream objects which are not permanently deleted.

25 16. The method of claim 15, wherein in the step (a) the search information corresponding to a range to be permanently deleted is deleted.

30 17. The method of claim 15 or 16, wherein the step (b) further comprising sub-steps of:

(b1) deleting search information related to the stream object unit which can be permanently deleted;

35 (b2) updating contiguous time related to the last stream object unit of the preceding stream object on the boundary between the stream object corresponding to the range to be permanently deleted and the preceding stream object; and

(b3) updating contiguous time related to the first stream object unit of the following stream object on the boundary between the stream object corresponding to the range to be permanently deleted and the following stream object.

40 18. The method of claim 17, wherein the step (b2) further comprises the sub-steps of:

45 (b21) providing a first value obtained in the form of an integer by rounding up the arrival time of the first packet of the first stream object unit of the stream object corresponding to the range to be permanently deleted, and providing a second value obtained by taking the integer part of the arrival time of the last packet of the last stream object unit of the preceding stream object;

50 (b22) providing a first subtraction result by subtracting the second value from the first value, and providing a second subtraction result by subtracting the first subtraction result from contiguous time of the last stream object unit of the preceding stream object; and

(b23) providing the contiguous time related to the last stream object unit of the preceding stream object modified by incrementing the second subtraction result by the unit value.

55 19. The method of claim 17 or 18, wherein the step (b3) further comprises the sub-steps of:

(b31) determining whether or not the arrival time of the first packet of the first stream object unit of the following stream object; and

(b32) when the arrival time of the first packet of the first stream object unit of the following stream object is not an integer, modifying contiguous time of the first stream object unit by incrementing it by the unit value, and otherwise, providing the contiguous time of the first stream object of the following stream object without change.

- 5 20. A recording medium storing search information for searching a plurality of stream objects, each of which has additional information on the arrival time of packet data, and link information for linking the search information to the actual stream objects and storing a first additional information (TE) indicating that temporary deletion was performed in the stream object corresponding to the temporary deletion range, and a second additional information (CT) indicating that the stream object including the first additional information and the following stream object were
10 one contiguous stream object before temporary deletion.
21. The recording medium of claim 20, wherein search information and link information for a plurality of stream objects generated corresponding to ranges to be deleted are updated, and the first additional information and the second additional information are included in the link information.
- 15 22. The recording medium of claim 20, wherein search information and link information for a plurality of stream objects generated corresponding to ranges to be deleted are updated, and the first additional information and the second additional information are included in the search information.

20

25

30

35

40

45

50

55

FIG. 1

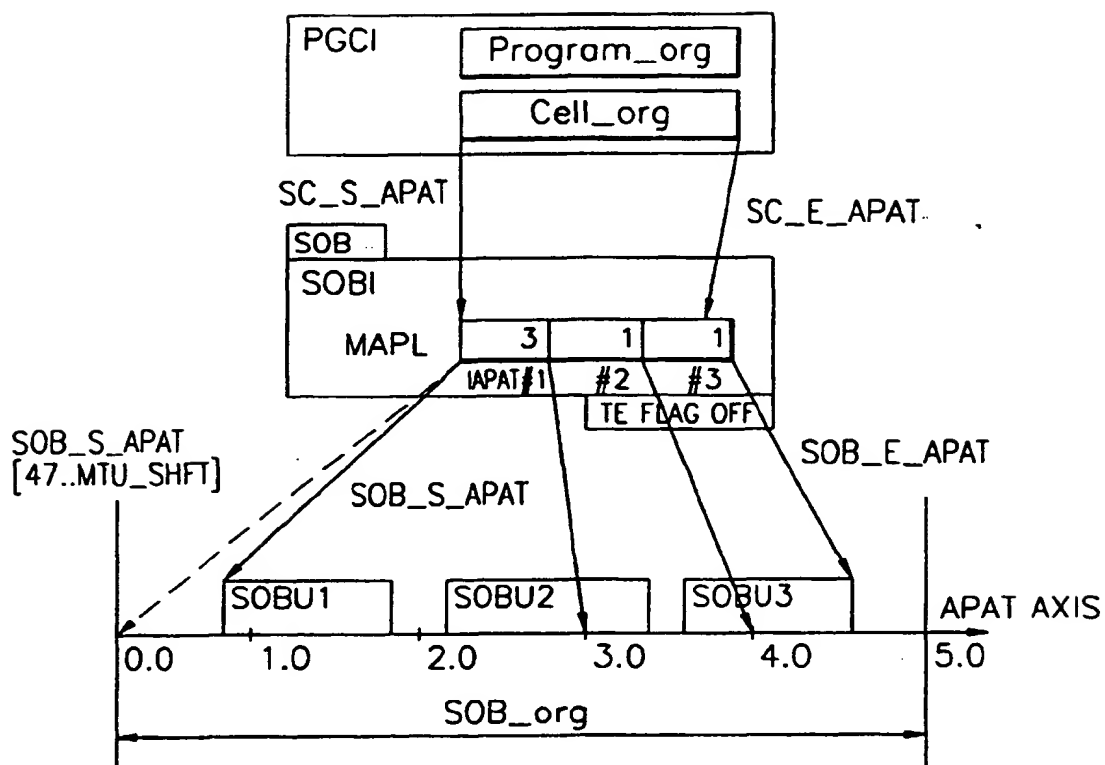


FIG. 2

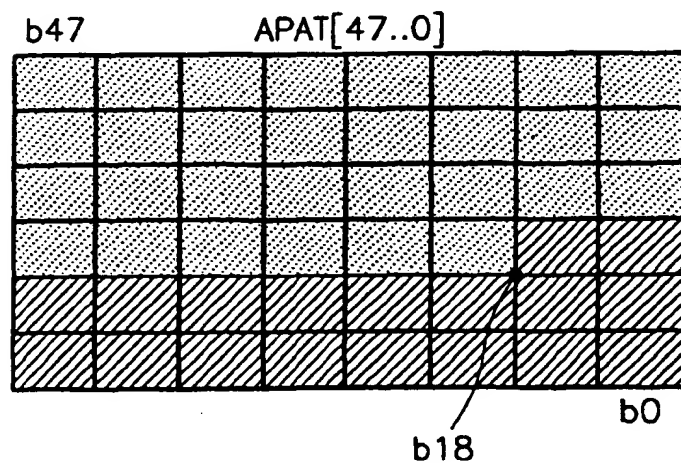


FIG. 3

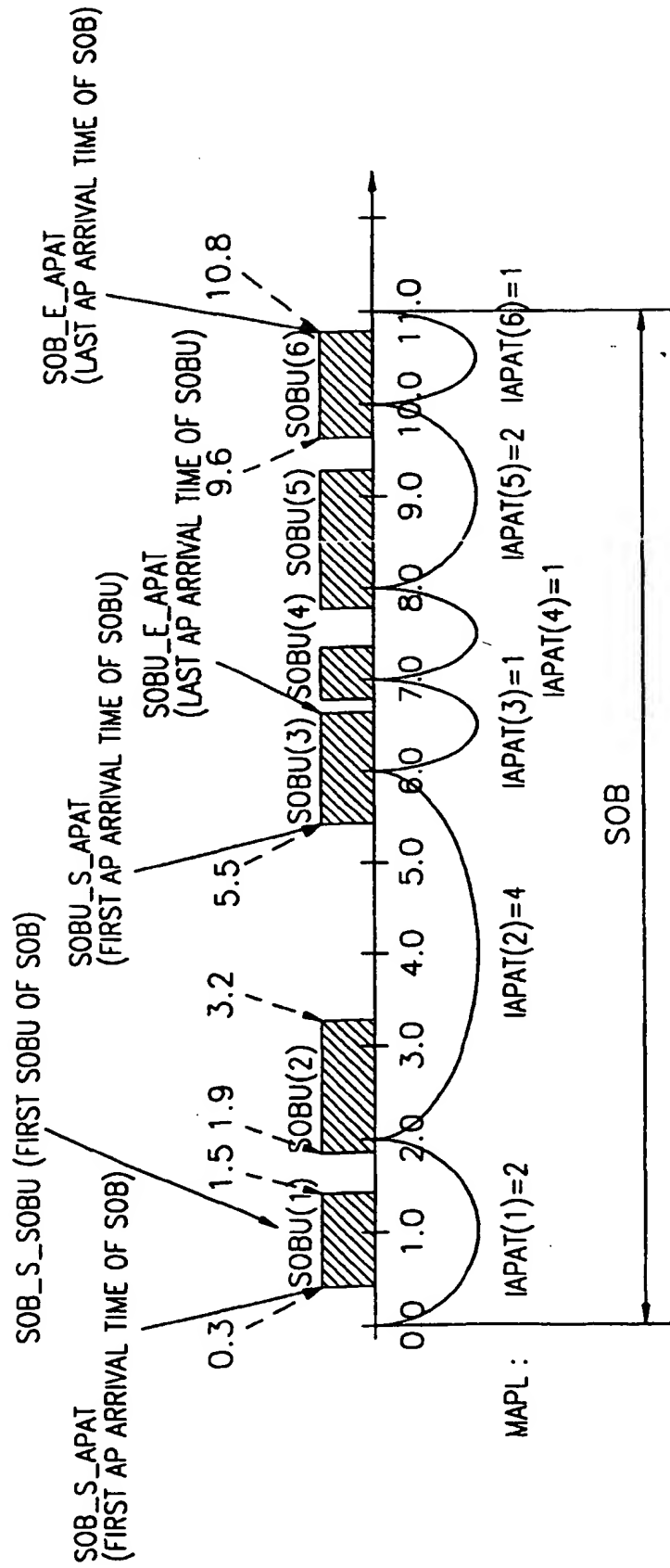


FIG. 4A

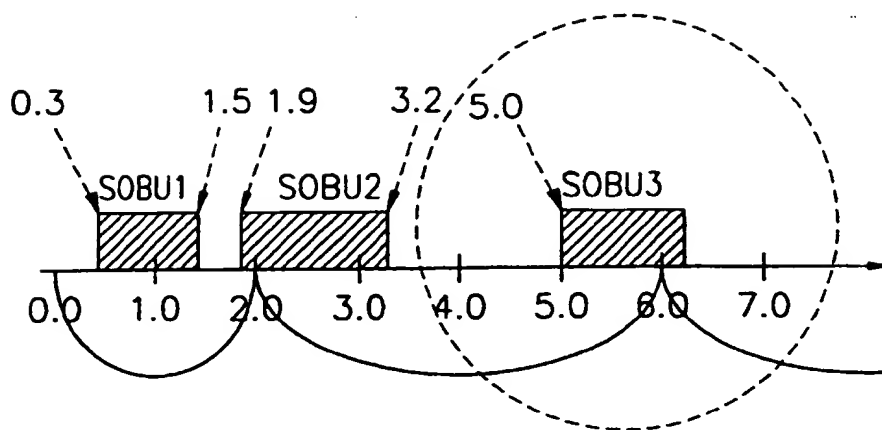


FIG. 4B

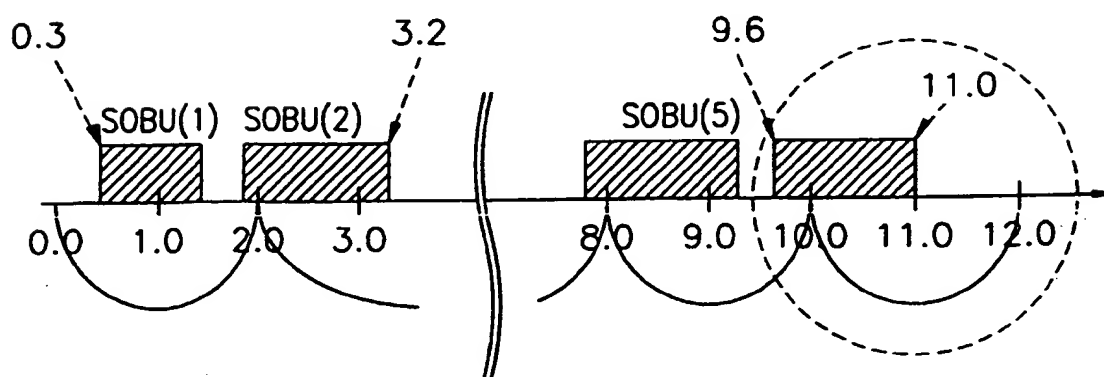


FIG. 5

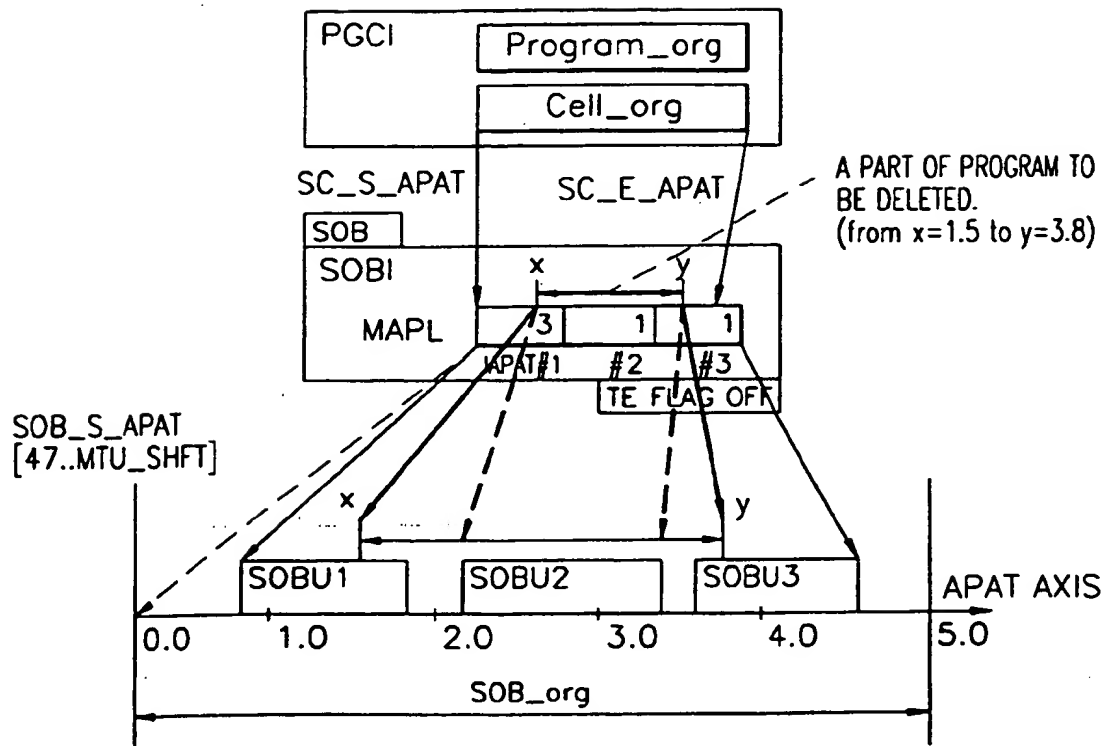


FIG. 6

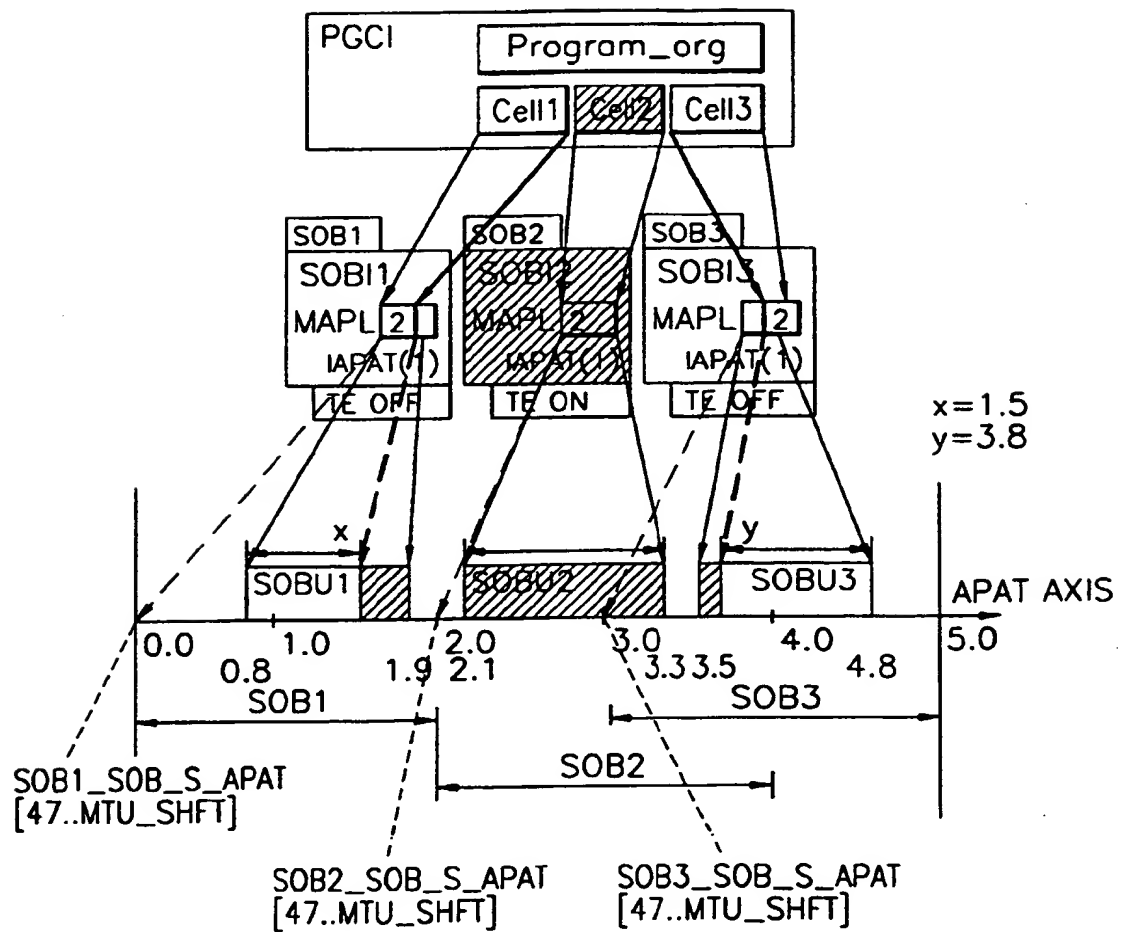


FIG. 7

	SOB_S _APAT	SOB_E _APAT	SOB_S _SOBU	MAPL_ ENT_Ns	MAPL [IAPATs]	SOB_TY	SC_S_ APAT	SC_E_ APAT
BEFORE TEMPORAL DELETION, SOB	0.8	4.5	SOBU1	3	[3,1,1]		0.8	4.5
AFTER TEMPORAL DELETION, SOB1	0.8	1.9	SOBU1	1	[2]		0.8	1.5
SOB2	2.1	3.3	SOBU2	1	[2]	TE,CT	2.1	3.3
SOB3	3.5	4.8	SOBU3	1	[2]	CT	3.8	4.5

FIG. 8

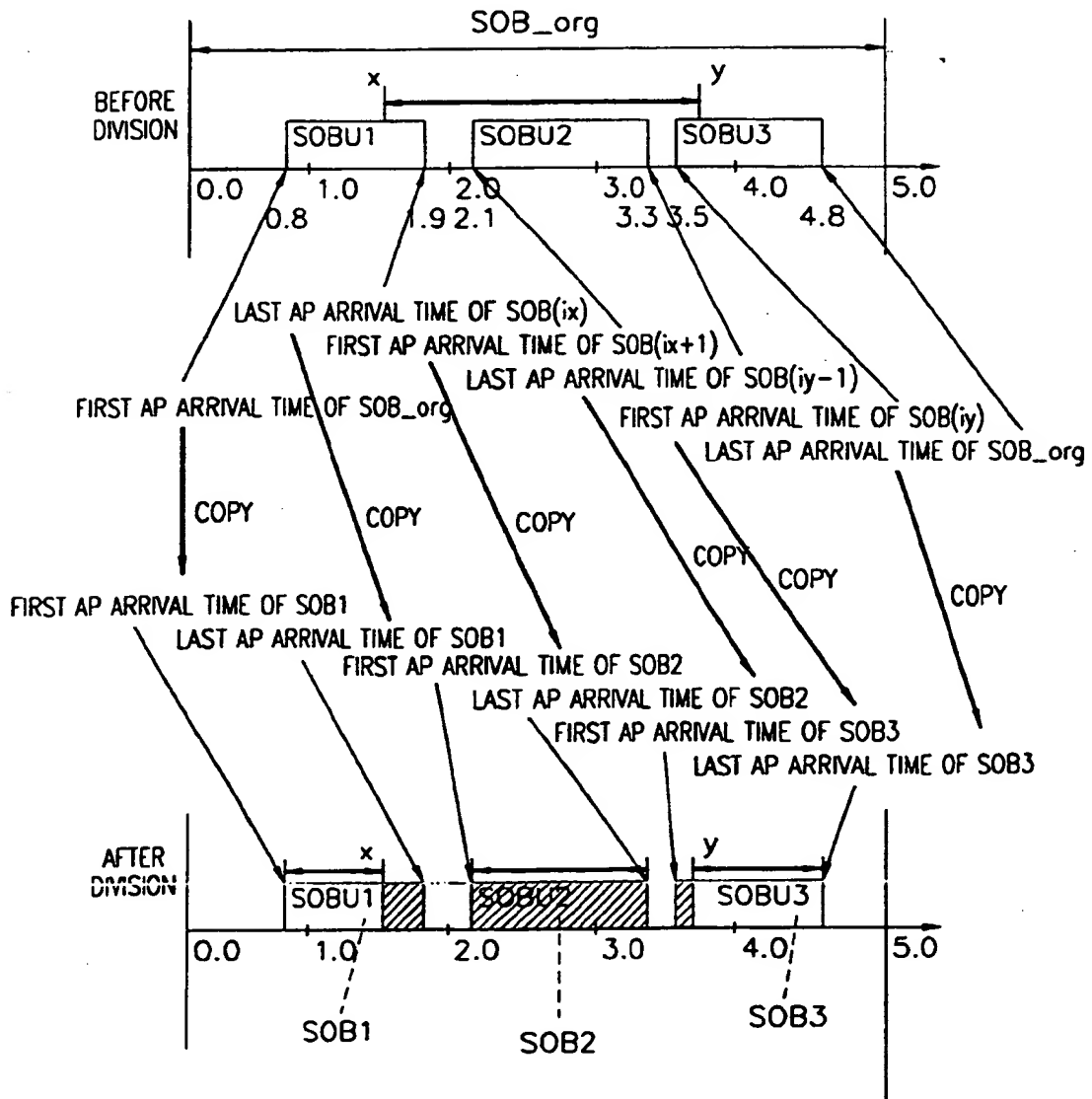


FIG. 9

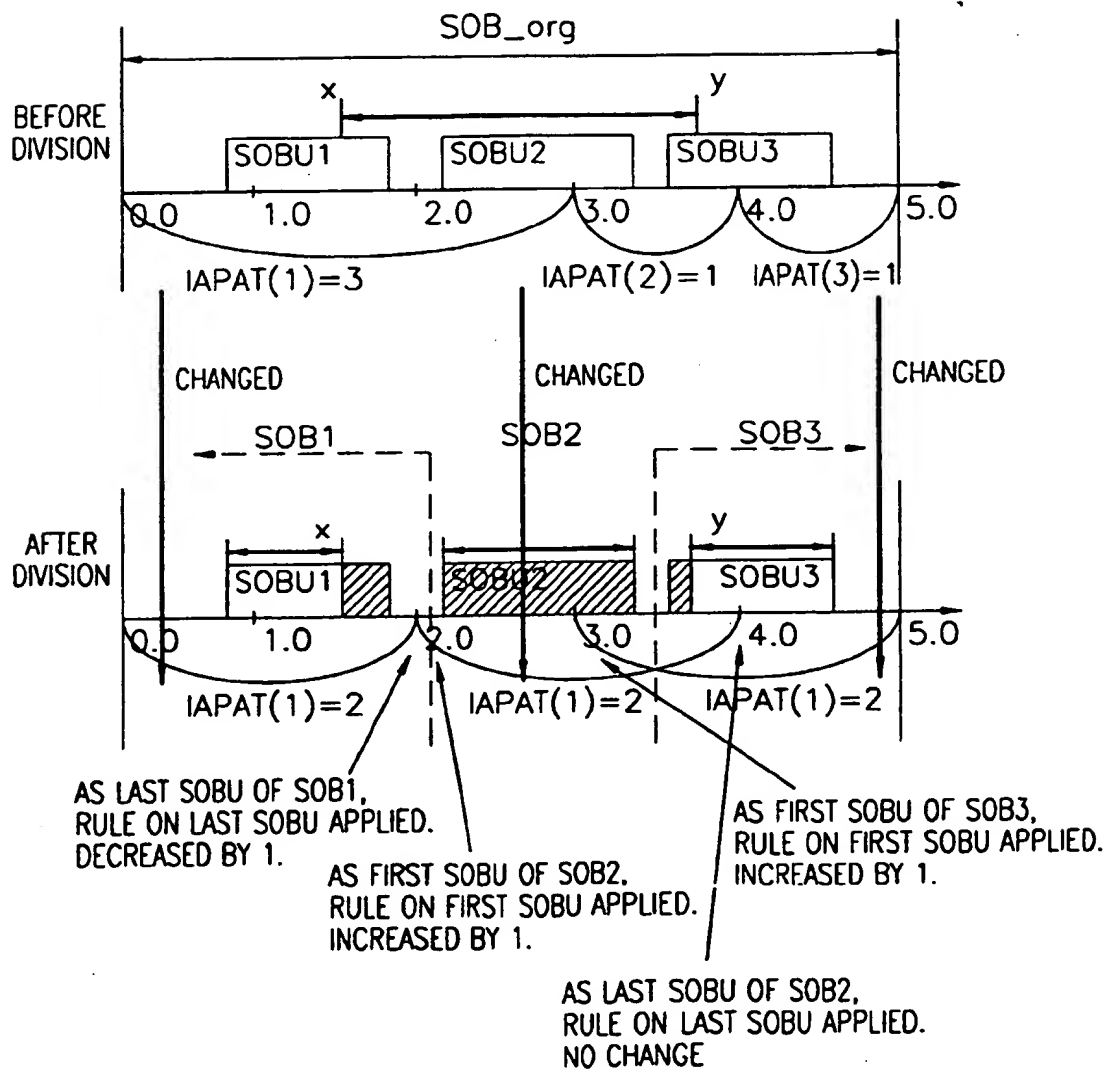


FIG. 10A

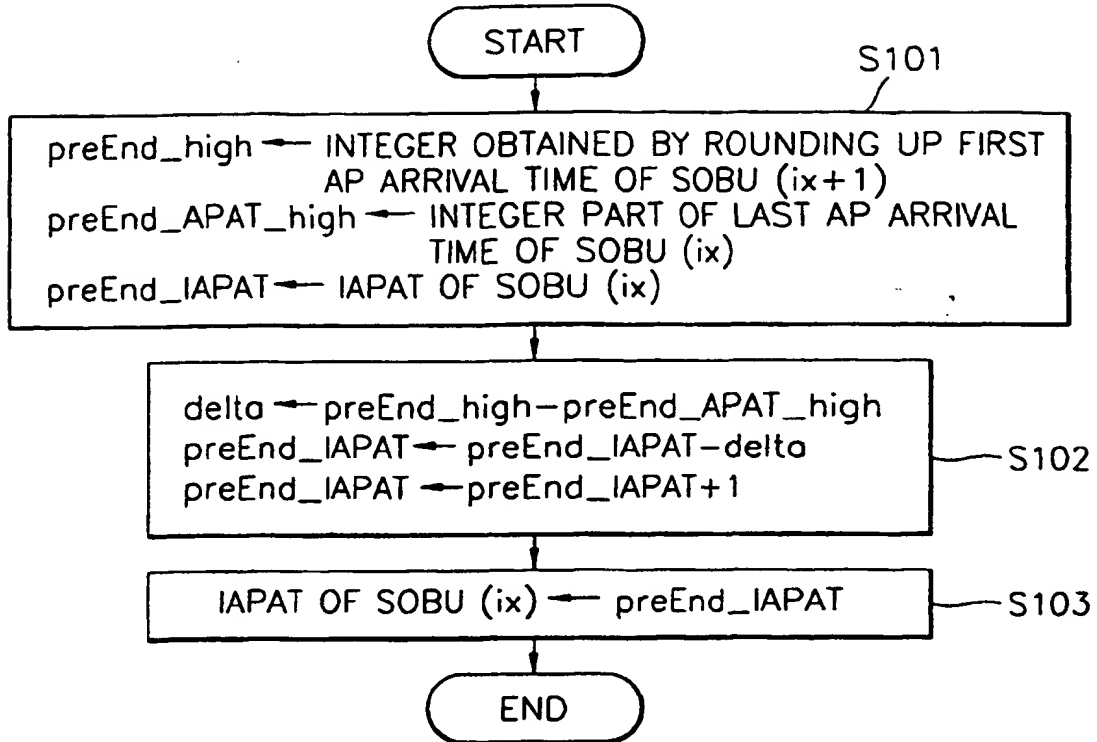


FIG. 10B

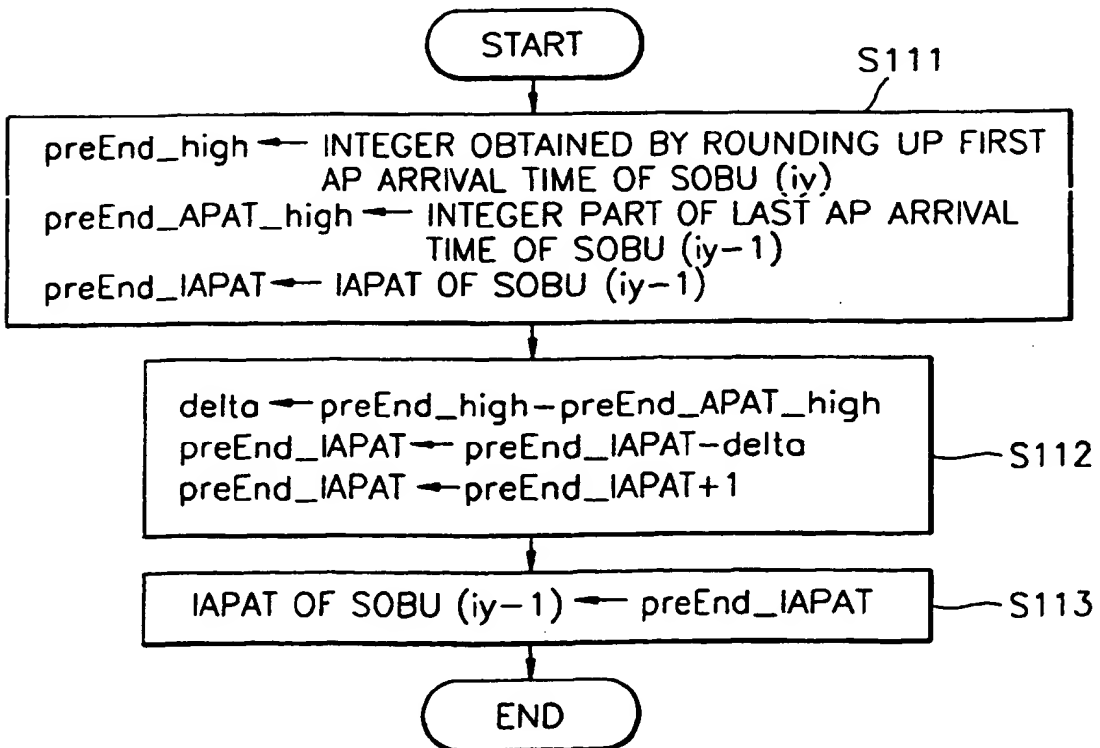


FIG. 11A

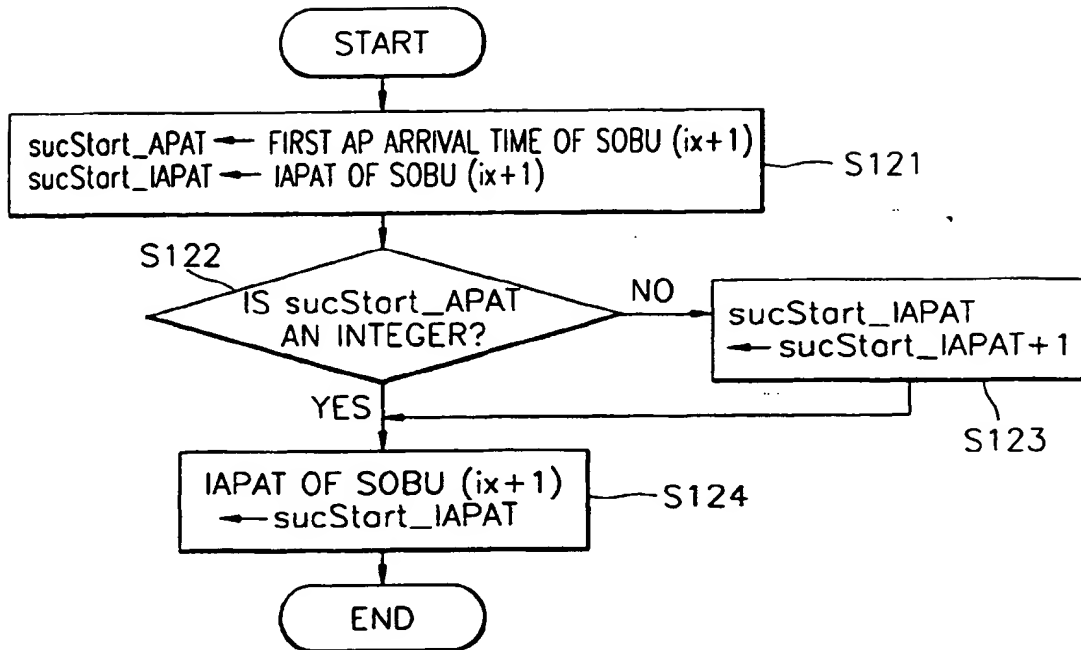


FIG. 11B

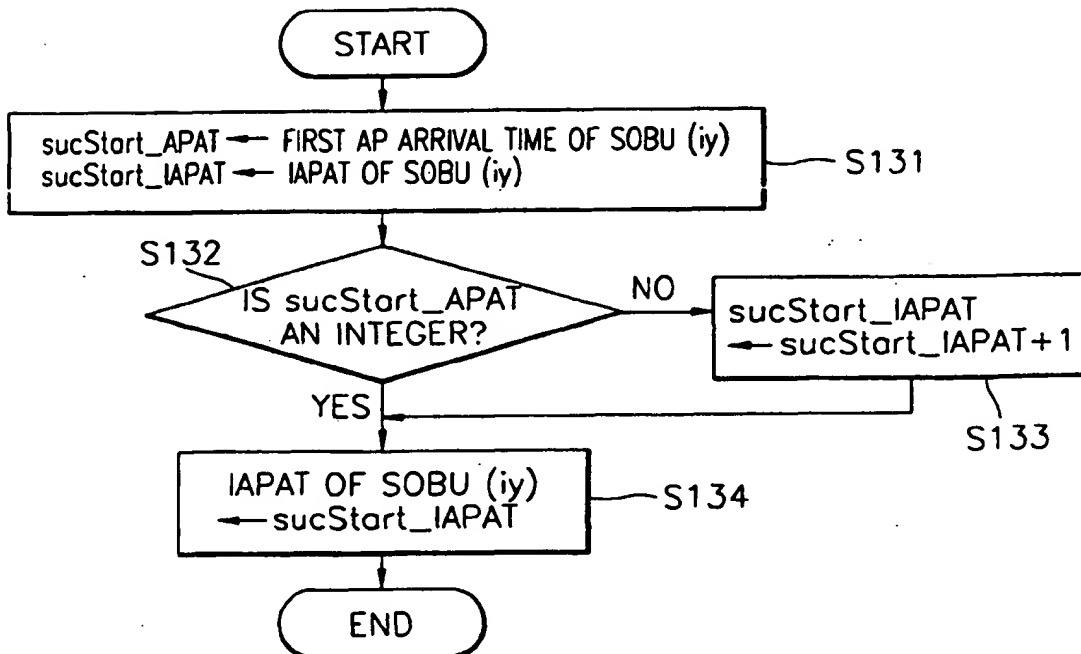


FIG. 12

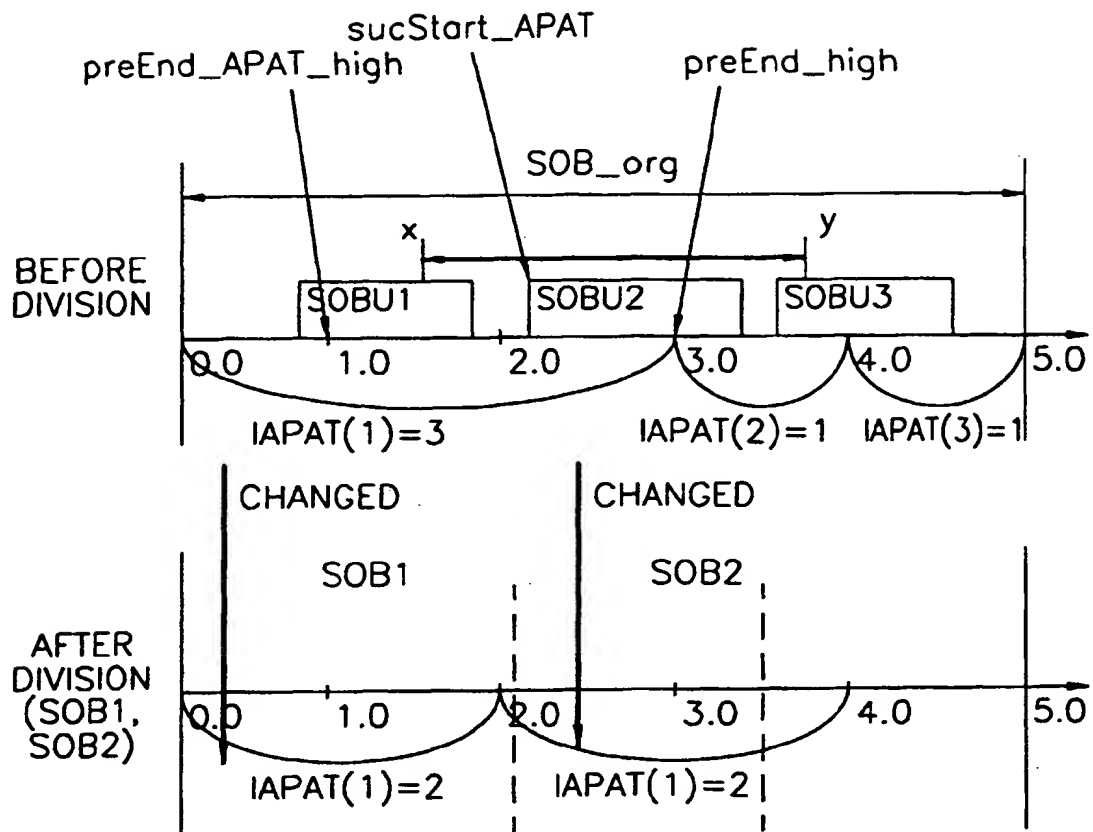


FIG. 13

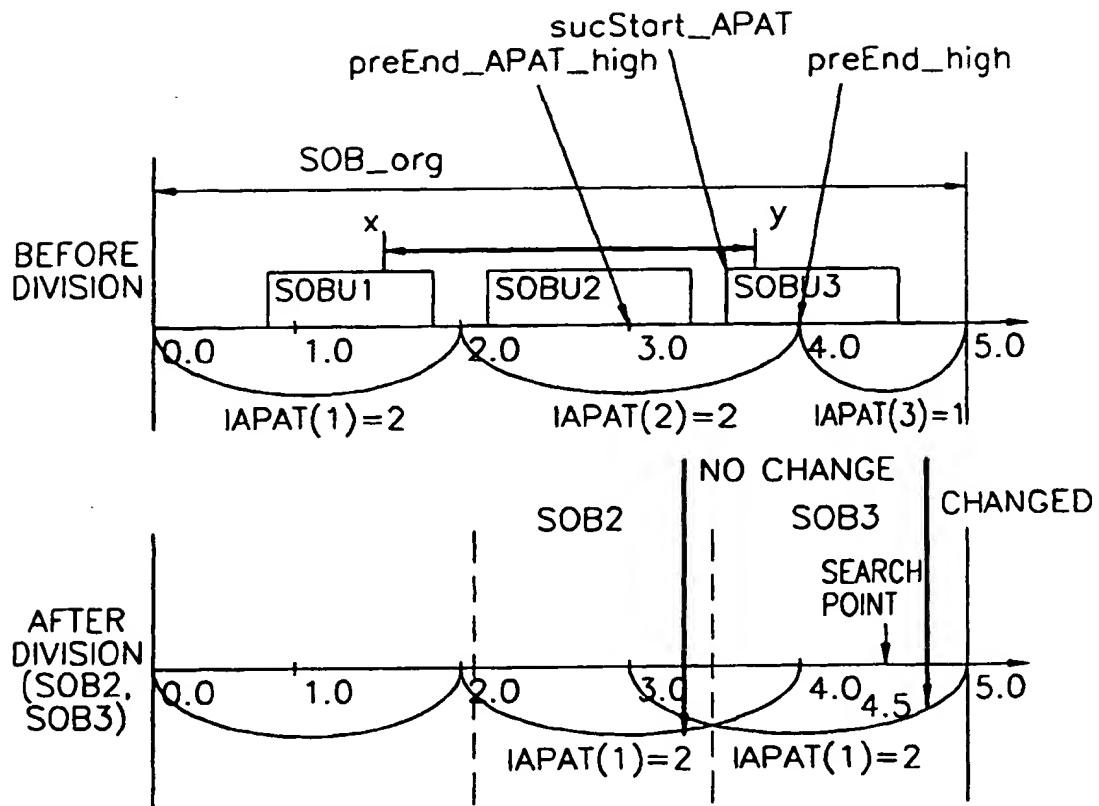


FIG. 14

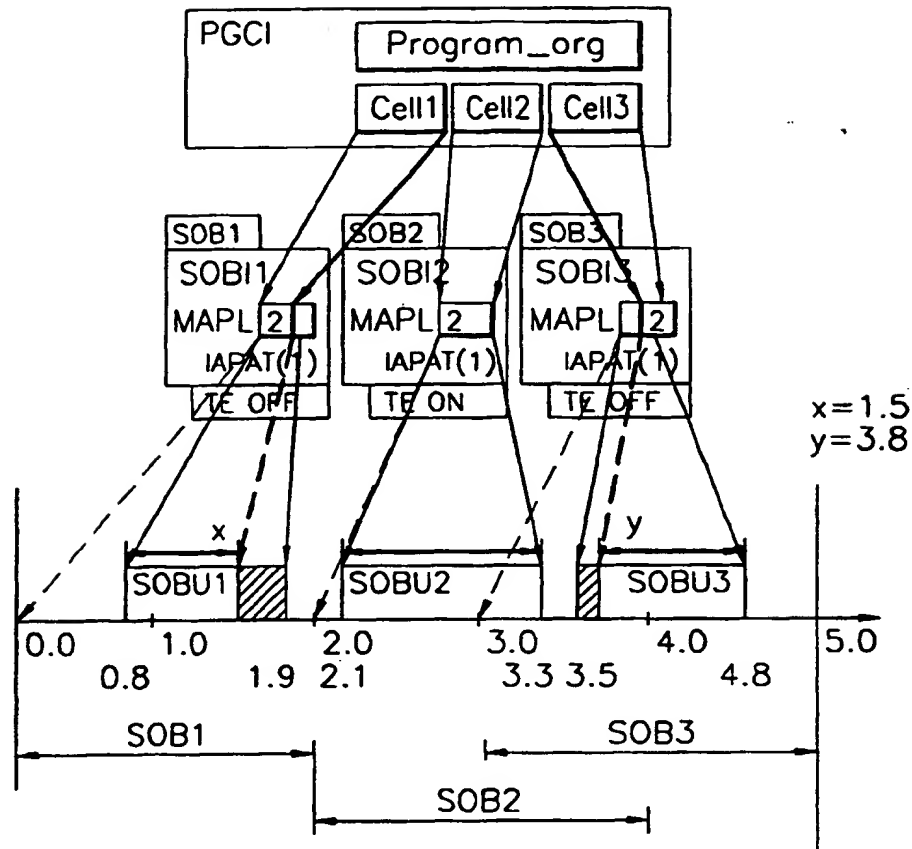


FIG. 15

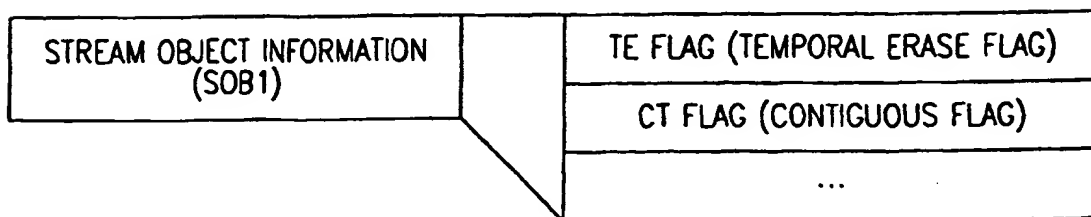


FIG. 16

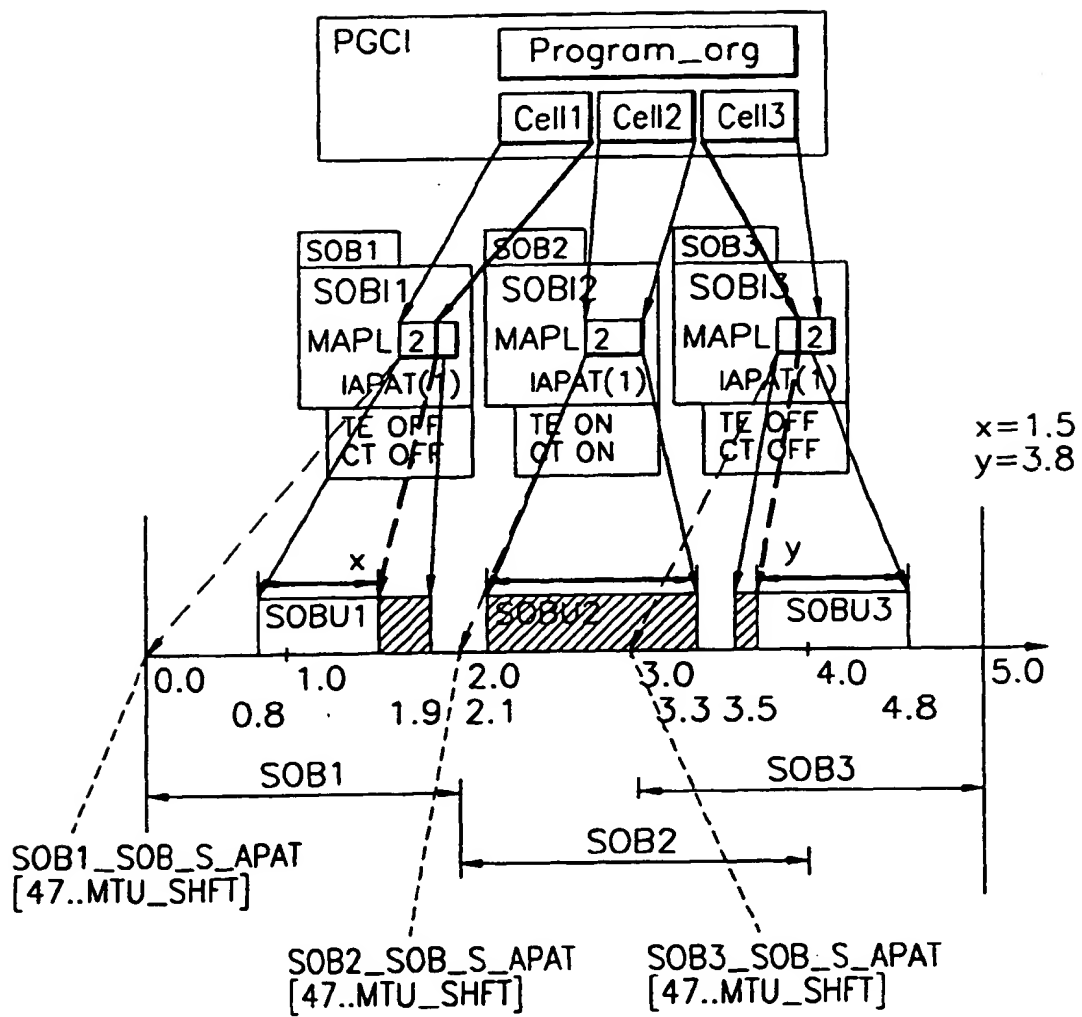


FIG. 17

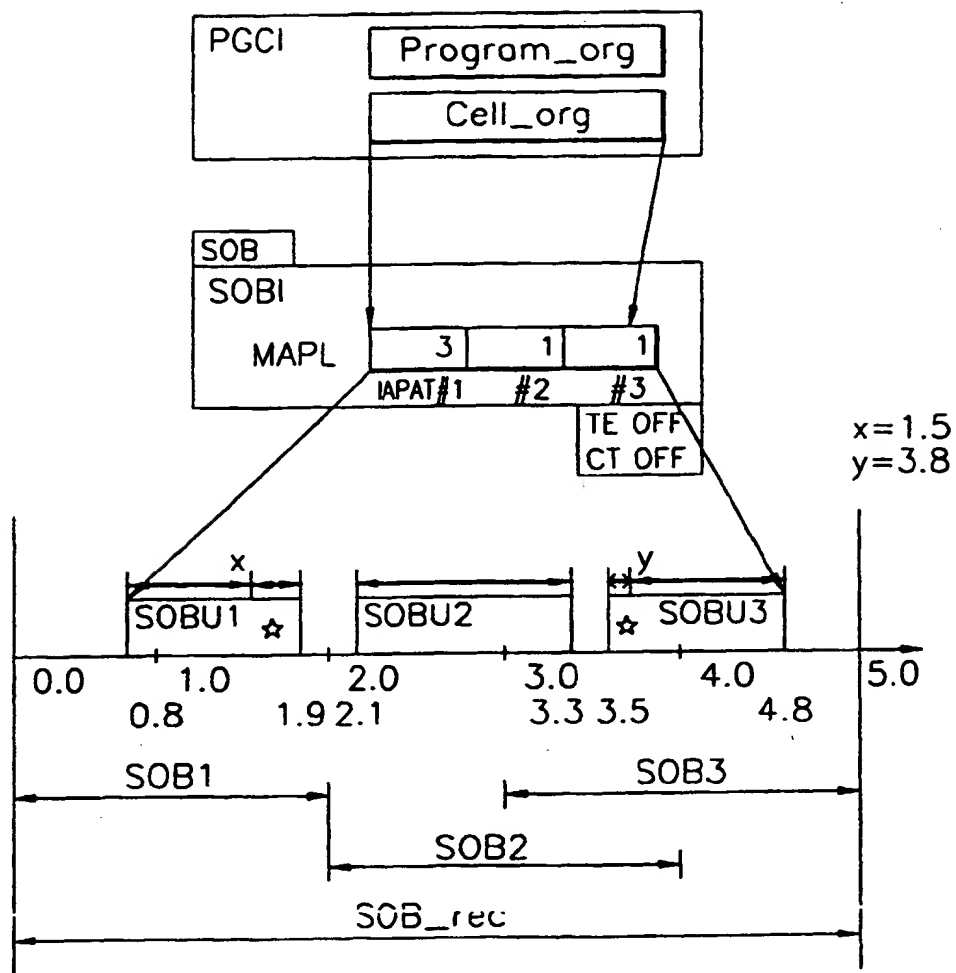


FIG. 18

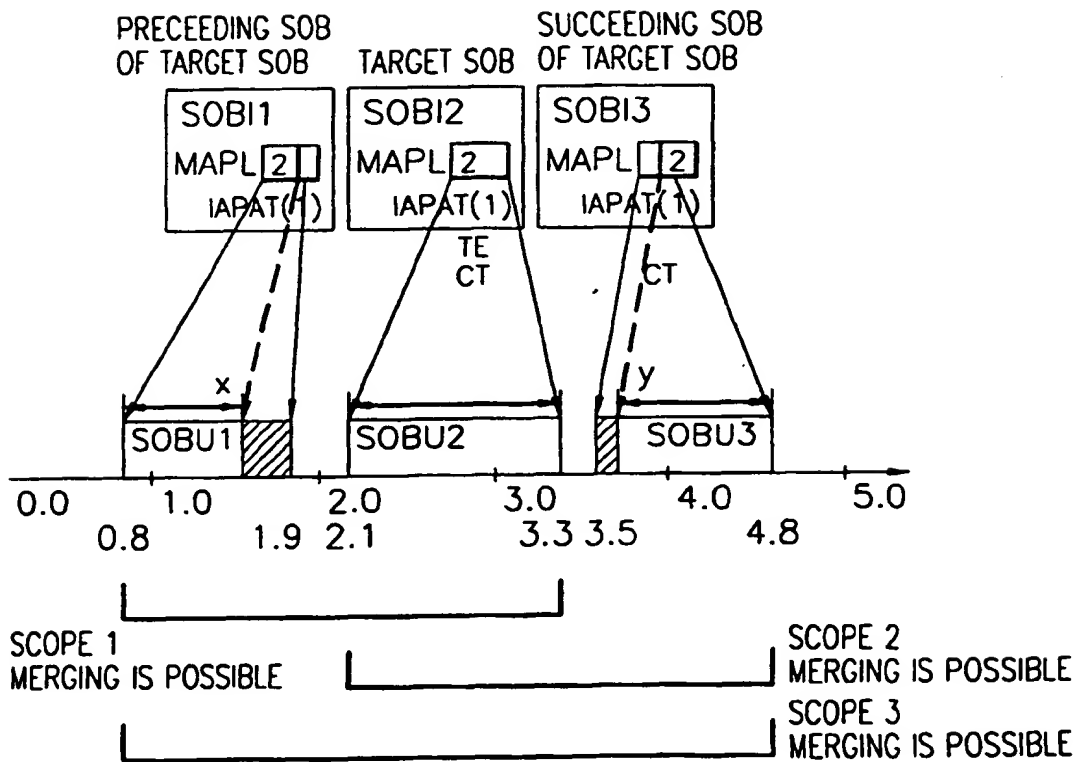


FIG. 19

		SOB_S _APAT	SOB_E _APAT	SOB_S _SOBU	MAPL_ ENT_Ns	MAPL [IAPAT's]	SOB_TY	SC_S_ APAT	SC_E_ APAT
BEFORE TEMPORAL DELETION	SOB_ org	0.8	4.5	SOBU1	3	[3,1,1]		0.8	4.5
AFTER TEMPORAL DELETION	SOB1	0.8	1.9	SOBU1	1	[2]		0.8	<u>1.5</u>
	SOB2	2.1	3.3	SOBU2	1	[2]	TE,CT	2.1	3.3
	SOB3	3.5	4.5	SOBU3	1	[2]	CT	<u>3.8</u>	4.5
AFTER RESTORATION	SOB_ rec	0.8	4.5	SOBU1	3	[3,1,1]		0.8	4.5

FIG. 20

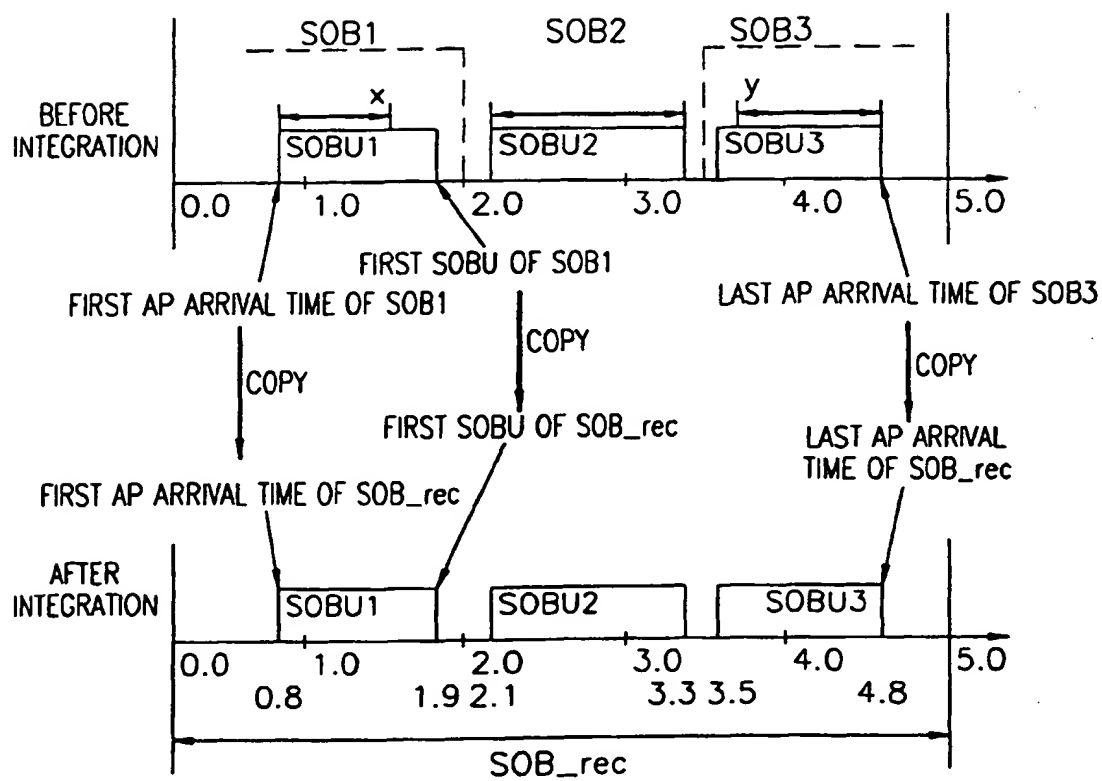


FIG. 21

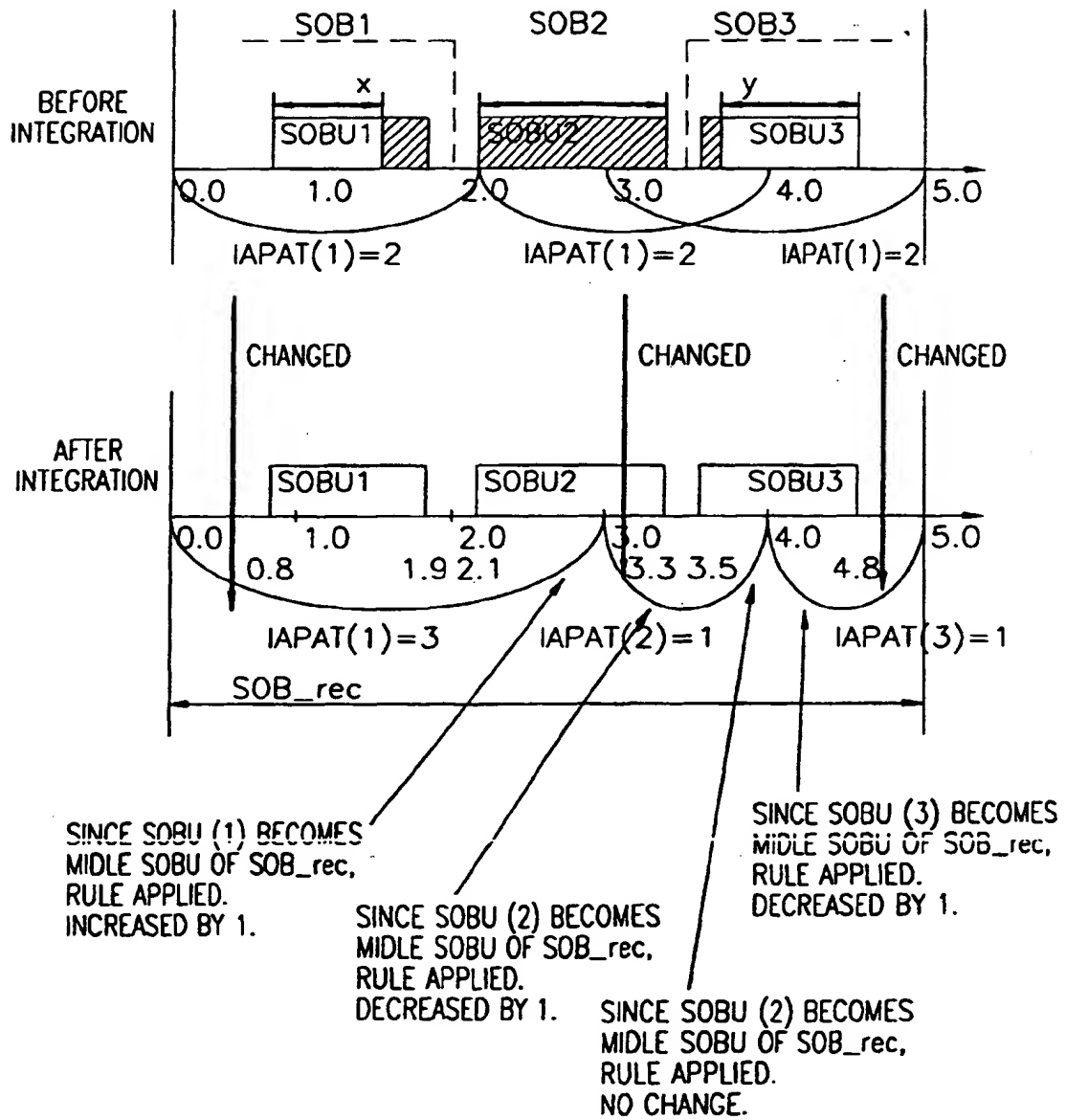


FIG. 22A

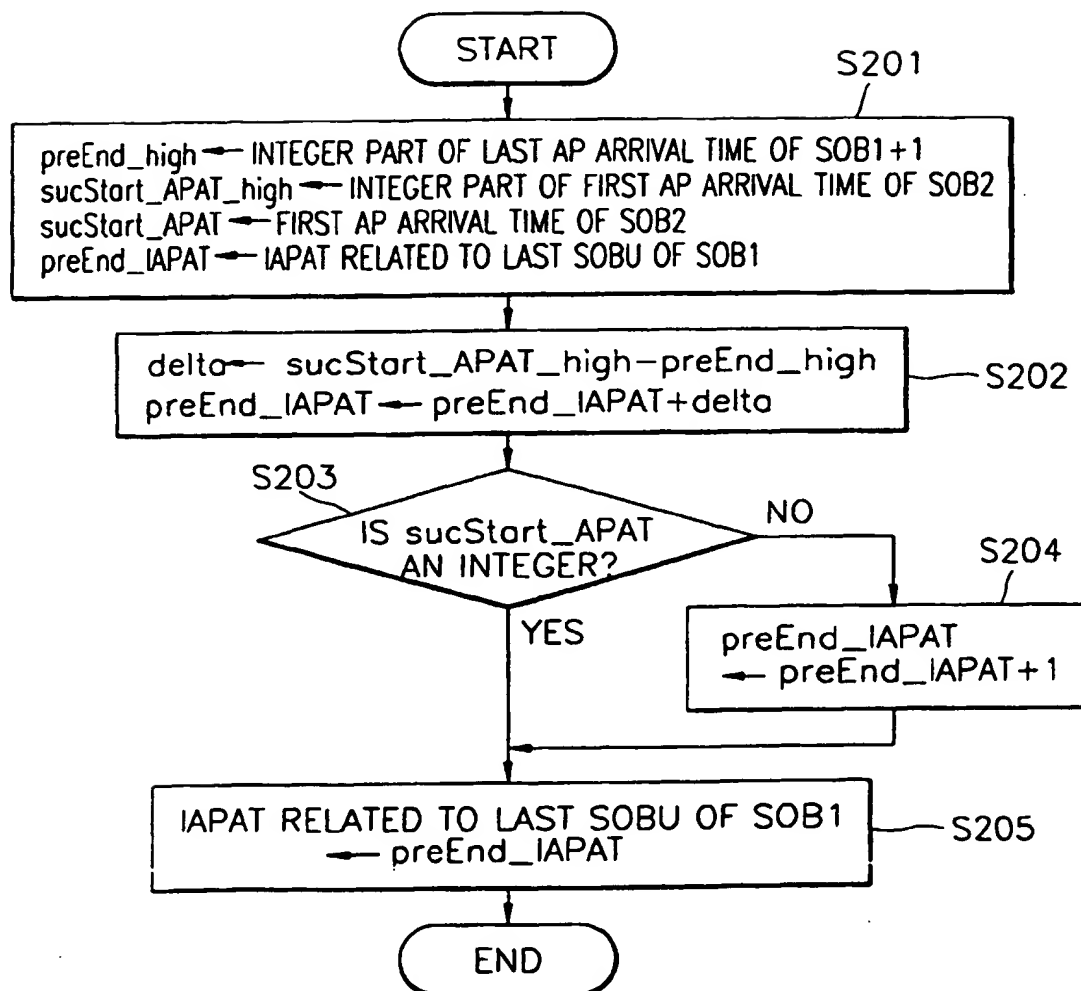


FIG. 22B

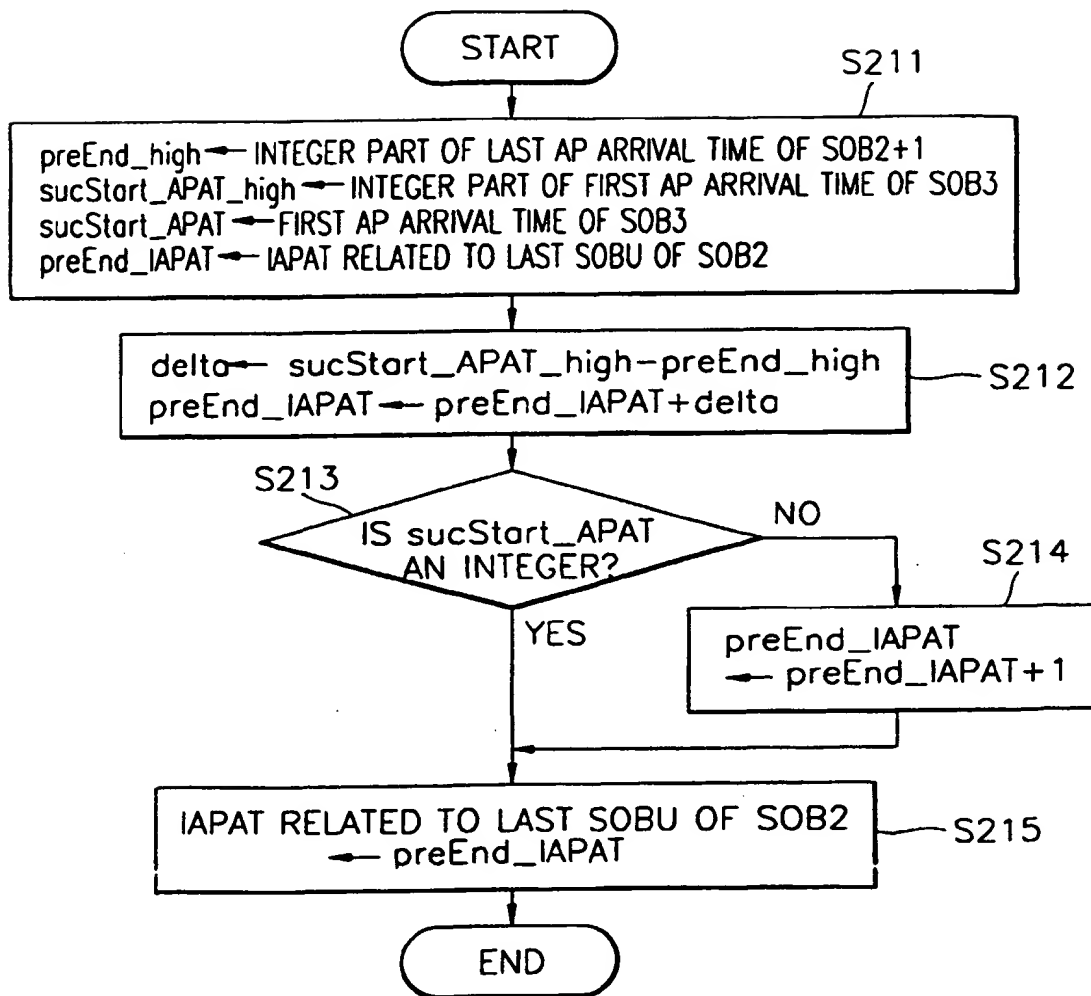


FIG. 23A

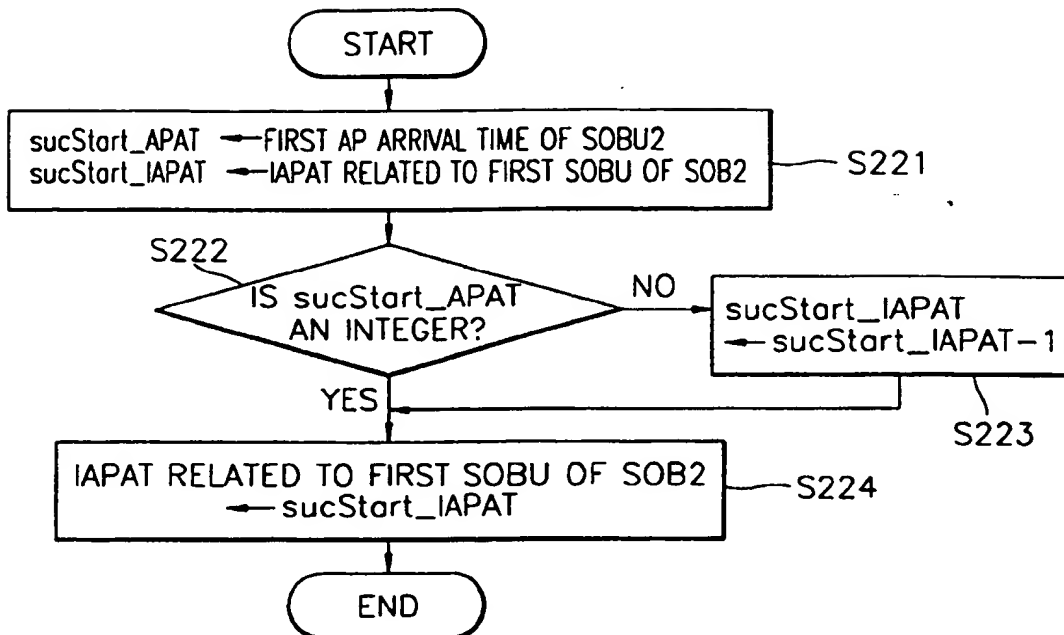


FIG. 23B

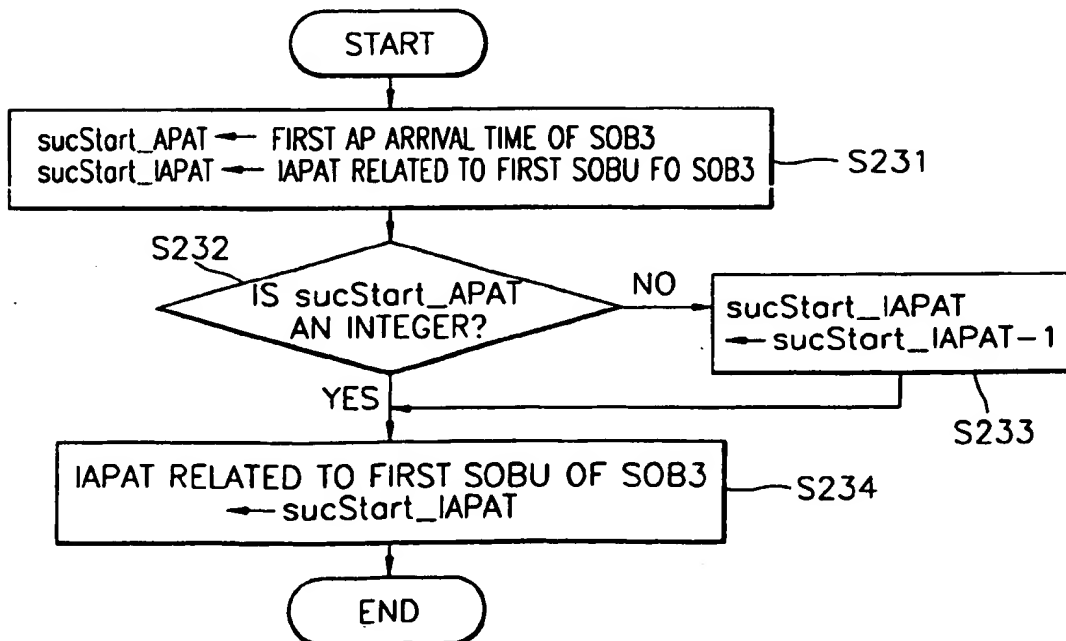


FIG. 24

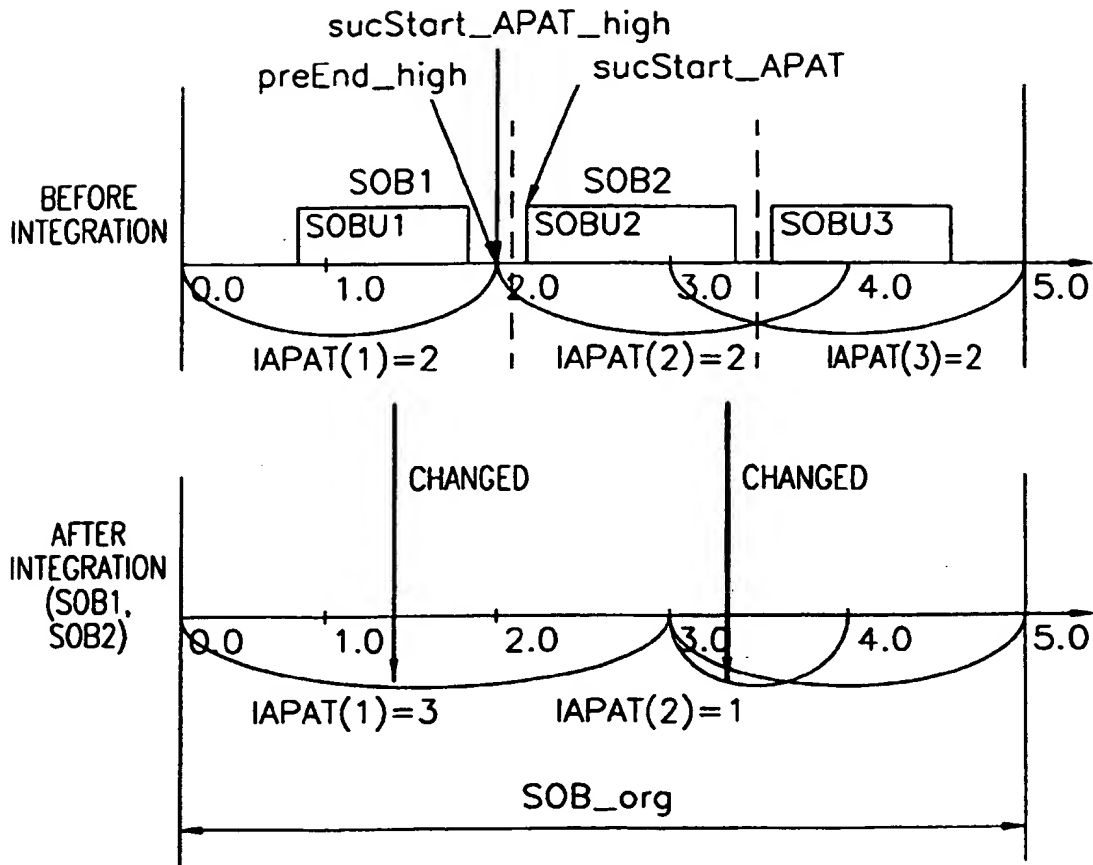


FIG. 25

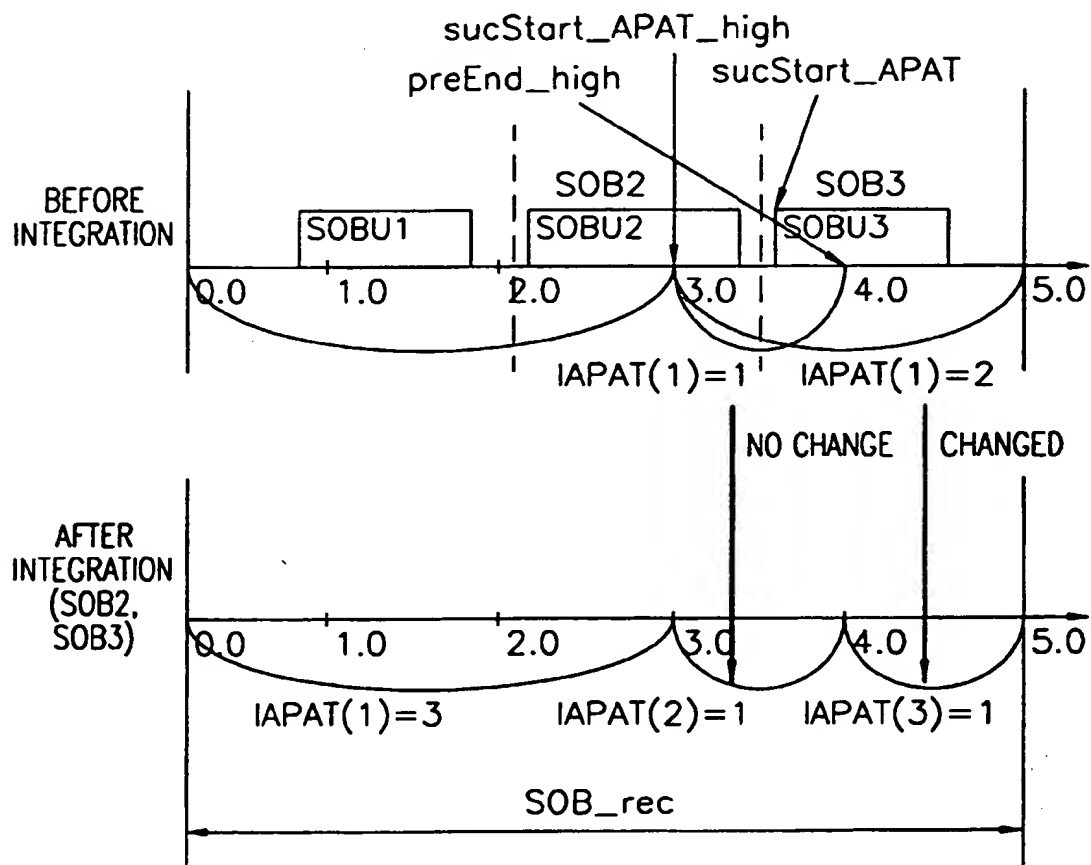


FIG. 26

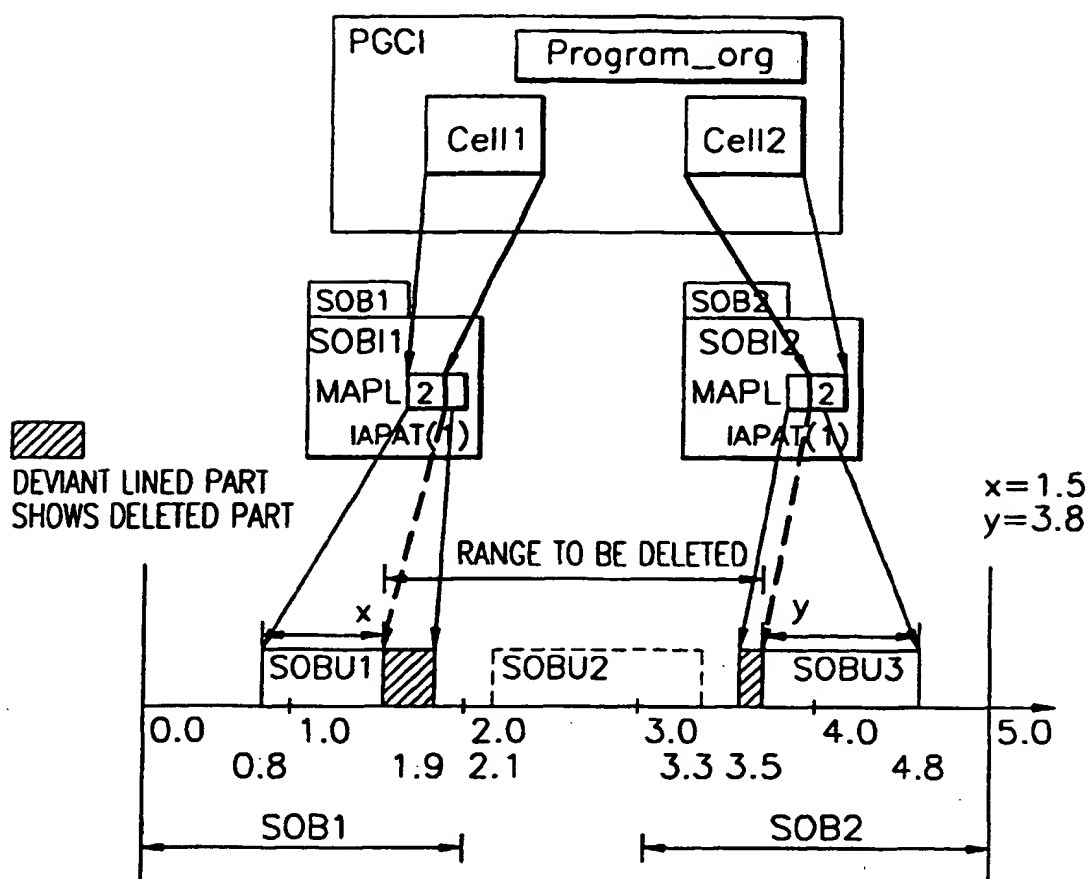


FIG. 27

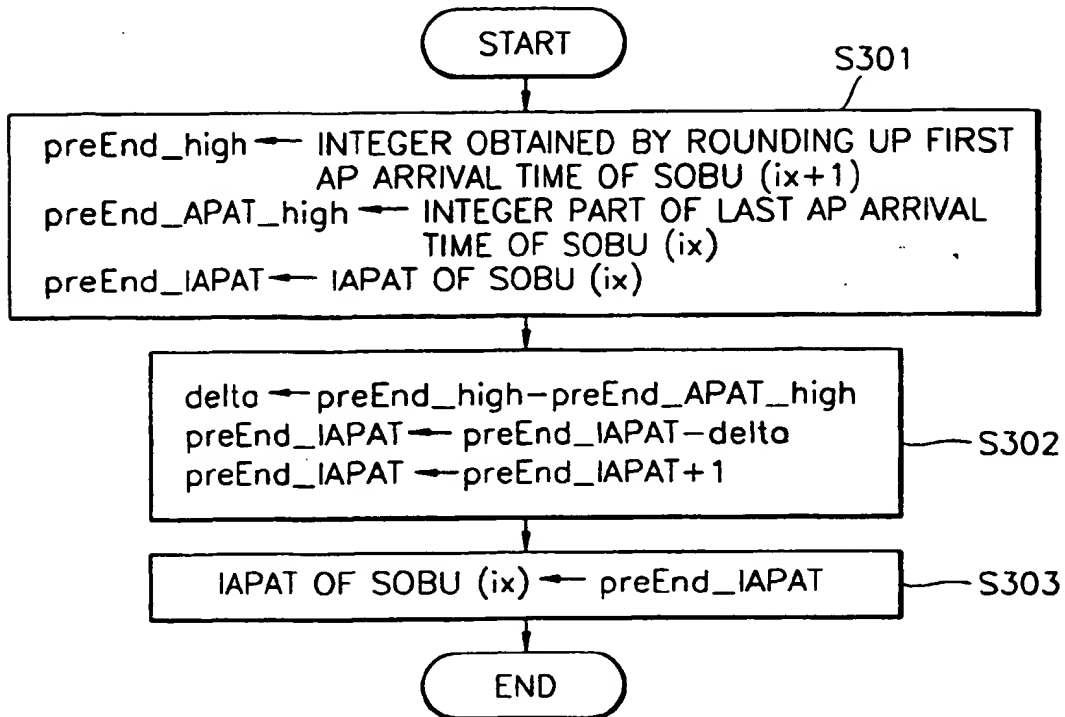
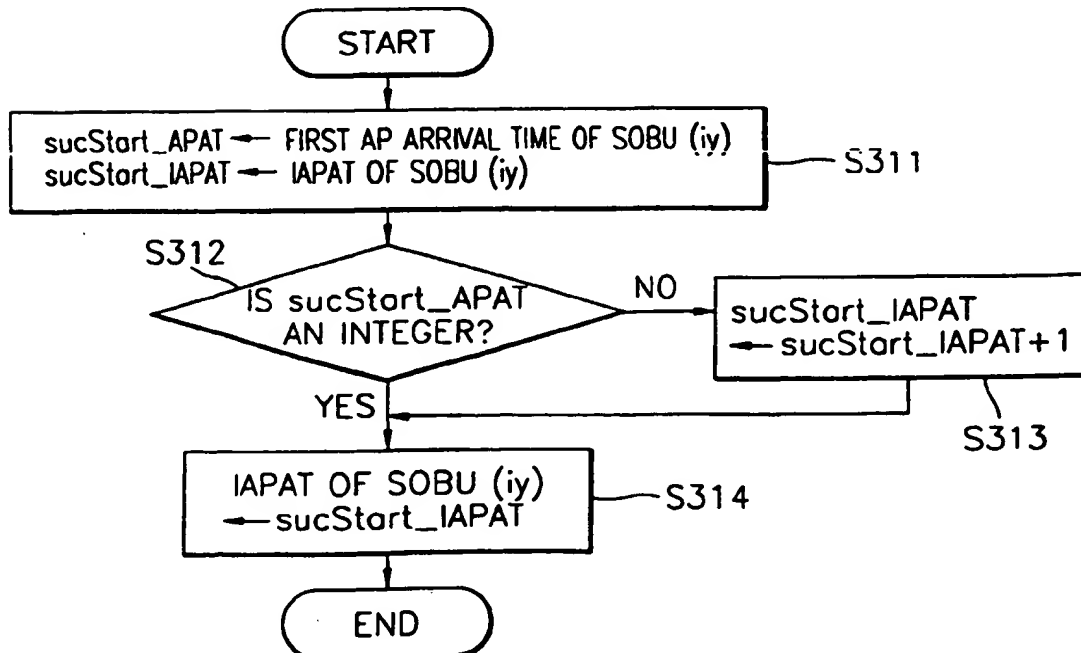


FIG. 28





European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 30 3506

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
E	EP 1 021 048 A (TOKYO SHIBAURA ELECTRIC CO) 19 July 2000 (2000-07-19) * page 3, line 57 - page 7, line 17 * * page 18, line 26-54 * ---	1,20	G11B27/034 G11B20/12 H04N5/85 G11B27/32
P,X	EP 0 986 062 A (THOMSON BRANDT GMBH) 15 March 2000 (2000-03-15) * the whole document * ---	5	
P,A	WO 99 38166 A (UNO TOHRU ;ISHIZAWA YOSHIYUKI (JP); TOKYO SHIBAURA ELECTRIC CO (JP) 29 July 1999 (1999-07-29) * page 17, line 11 - page 34, line 17 * ---	1,6,9, 12,15	
A	EP 0 833 337 A (SONY CORP) 1 April 1998 (1998-04-01) * the whole document * ---	1,6,9, 12,15	
A	EP 0 903 744 A (MATSUSHITA ELECTRIC IND CO LTD) 24 March 1999 (1999-03-24) * column 19, line 38 - column 44, line 26 * -----	1,6,9, 12,15	
			TECHNICAL FIELDS SEARCHED (Int.Cl.7) G11B H04N
The present search report has been drawn up for all claims			
Place of search THE HAGUE		Date of completion of the search 3 August 2000	Examiner Mourik, J
<p>CATEGORY OF CITED DOCUMENTS</p> <p>X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document</p> <p>T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document</p>			

**ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO.**

EP 00 30 3506

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

03-08-2000

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 1021048 A	19-07-2000	NONE	
EP 0986062 A	15-03-2000	EP 0991072 A	05-04-2000
		AU 5742899 A	27-03-2000
		WO 0014744 A	16-03-2000
		AU 5854999 A	27-03-2000
		WO 0014743 A	16-03-2000
WO 9938166 A	29-07-1999	EP 0970473 A	12-01-2000
		JP 11317056 A	16-11-1999
EP 0833337 A	01-04-1998	JP 10106237 A	24-04-1998
		US 6088304 A	11-07-2000
EP 0903744 A	24-03-1999	EP 1020862 A	19-07-2000
		WO 9914935 A	25-03-1999
		JP 11187354 A	09-07-1999

THIS PAGE BLANK (USPTO)